Proyecto Fin de Carrera
Ingeniería Electrónica, Robótica y Mecatrónica

Neuroestimulador de ondas arbitrarias

Autor: Pablo Jiménez Fernández
Tutor: Hipólito Guzmán Miranda
Alejandro Barriga-Rivera

Dpto. de Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2019

GRUPO DE INGENIERÍA ELECTRÓNICA

Proyecto Fin de Carrera
Ingeniería Electrónica, Robótica y Mecatrónica

Neuroestimulador de ondas arbitrarias

Autor:
Pablo Jiménez Fernández

Tutores:
Hipólito Guzmán Miranda
Profesor contratado doctor
Alejandro Barriga-Rivera
Research fellow

Dpto. de Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2019

Proyecto Fin de Carrera: Neuroestimulador de ondas arbitrarias

Autor: Pablo Jiménez Fernández

Tutor: Hipólito Guzmán Miranda
Alejandro Barriga-Rivera

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Sevilla, 2019

A mi familia

A mis maestros

Agradecimientos

A mis tutores de proyecto, Alejandro Barriga-Rivera e Hipólito Guzmán, por brindarme esta magnífica oportunidad de la que tanto he apredido y disfrutado. Gracias por la incansable predisposición, paciencia y por querer transmitirme vuestra sabiduría.

A mis padres y a mi hermano, por el incondicional apoyo que me han ofrecido en todo momento.

A Marta, por creer en mí.

Pablo Jiménez Fernández

Sevilla, 2019

Resumen

Un neuroestimulador es un dispositivo médico capaz de suministrar corriente controlada al tejido excitable objetivo. Al hacerlo, se puede lograr la modulación de la actividad del sistema nervioso. La técnica habitual más empleada se basa en el envío de pulsos de carga bifásica y balanceada. Este método permite la entrega y recogida de la misma cantidad de carga eléctrica, evitando dañar el tejido sobre el que se aplican los estímulos y ralentizar la degradación de los electrodos.

El objetivo de este diseño es fabricar un estimulador neuronal que pueda usarse en estudios preclínicos de forma segura. El estimulador neuronal proporcionará capacidades para estimular simultáneamente al menos a través de dos canales independientes. Este dispositivo debe ser programable y capaz de entregar las formas de onda de estímulo deseadas a la llegada de una señal de disparo. El sistema estará eléctricamente aislado de la fuente de alimentación por seguridad eléctrica. La corriente inyectada debe ser precisa y capaz de proporcionar formas de onda de al menos 30 kHz.

Este dispositivo eventualmente ayudará a la investigación de nuevos paradigmas de estimulación que contribuyan, entre otros, a mejorar el rendimiento actual del ojo biónico.

Abstract

A neurostimulator is a medical device capable of delivering controlled current to target excitable tissue. By doing so, modulation of the activity of the nervous system can be achieved. The most commonly used technique is based on sending biphasic and balanced load pulses. This method allows the delivery and collection of the same amount of electrical charge, avoiding damaging the tissue on which the stimuli are applied and slowing down the degradation of the electrodes.

The aim of this design is to create a neuronal stimulator that can be safely used in preclinical studies. The neural stimulator will provide the ability to stimulate simultaneously through at least two independent channels. This device must be programmable and capable of delivering the desired stimulus waveforms at the arrival of a trigger signal. The system will be electrically isolated from the power supply for electrical safety. The injected current must be accurate and capable of providing waveforms of at least 30 kHz.

This device will eventually help the investigation of new stimulation paradigms that contribute, among others, to improve the current performance of the bionic eye.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

En la actualidad, el innegable avance tecnológico de la gran mayoría de ciencias nos informa sobre el creciente interés público por el desarrollo y mejora de las condiciones de vida de la sociedad. El campo de la ingeniería biomédica también se encuentra inmerso en este continuo progreso.

Las aplicaciones actuales para el tratamiento de patologías [1] [2] inducen la necesidad de un avance de las técnicas de neuroestimulación convencionales. Típicamente, se han empleado estímulos bifásicos de onda cuadrada sobre electrodos metálicos, inyectando y recogiendo la misma cantidad de corriente para evitar la degradación de los metales usados, consiguiendo así la activación de las células excitables.

Frente a lo convencional, se han consolidado nuevas estrategias de estimulación como, por ejemplo, estimulación no invasiva a través de campos eléctricos de interferencia temporal [3], acoplamientos inductivos [4] o estimulaciones de alta frecuencia en zonas de la médula dorsal [5].

De esta forma, para progresar en el terreno de la neuroestimulación, se hace necesario el uso de nuevas formas de onda más complejas, que permitan aprovechar los patrones de interferencias que se produzcan al estimular de manera simultánea en varias regiones.

El diseño aquí propuesto persigue permitir la ejecución controlada y automatizada de formas de onda de corriente arbitrarias hasta frecuencias de 30kHz. El sistema neuroestimulador implementado dispone de dos canales independientes y aislados entre sí, además de encontrarse aislado de la fuente de alimentación por motivos de seguridad [6]. Para evitar el almacenamiento de corrientes indeseadas que conduzcan a la degradación de los electrodos, el sistema cortocircuitará los terminales de salida cuando finalice la ejecución de cada estímulo [7].

Este dispositivo permitirá, también, el control simultáneo de varios estimuladores conectados a un solo sistema Raspberry Pi, quien se encargará de controlarlos. De esta forma, pueden crearse granjas de neuroestimuladores que permitan la ejecución múltiple de diferentes estímulos, consiguiendo activar en conjunto las diferentes regiones nerviosas.

El objetivo de este proyecto es dar la posibilidad a cualquier persona de poder construirse su propio neuroestimulador de forma sencilla y económica, en comparación con las opciones disponibles del mercado, fomentando así el desarrollo de la electrónica libre.

2 ARQUITECTURA DE MODELO DE VISTAS 4+1

El modelo de vistas de la arquitectura 4+1, diseñado por Philippe Kruchten [8], ha sido empleado para describir el sistema neuroestimulador de ondas arbitrarias, ya que los subsistemas que lo componen permiten una descripción que se ajusta perfectamente a cada una de las vistas que componen el modelo. En las secciones provistas en este capítulo se detallará el desarrollo implementado en el subsistema software y hardware del sistema neuroestimulador, además de la relación funcional que comparten. Finalmente, en la última vista se expondrán diferentes casos de uso como posibles ejemplos de funcionamiento.

2.1. Vista Lógica

El dispositivo neuroestimulador que se ha desarrollado en el presente trabajo combina un entorno software y otro hardware para configurar un sistema software que permita la entrega controlada y automatizada de formas de corriente arbitrarias comprendidas en el rango de $\pm$3mA.

Las formas de onda de corriente podrán ser enviadas bajo el control de una señal de disparo (*trigger signal*) a través de dos canales independientes que permitirán almacenar hasta un máximo de 100 estímulos por canal. Este conjunto de estímulos podrá ser programado por la persona usuaria. Además, el sistema software también permite predefinir y controlar, si se desea, el tiempo de ejecución que separa a dos estímulos consecutivos (*inter stimuli time*).

El neuroestimulador podrá operar tras ser alimentado a una tensión igual a +5V, suministrada a través del zócalo *USB Type-A* provisto o a través de los pines definidos como '*+5V*' y '*GND*'. Así mismo, como requisito de seguridad eléctrica, el sistema estará aislado eléctricamente de la fuente de alimentación con el fin de evitar que se introduzcan en el sistema las posibles interferencias ocasionadas por la propia fuente de alimentación o la red eléctrica, además de evitar bucles de tierra que puedan alterar los patrones de estimulación. Del mismo modo, para evitar las posibles perturbaciones que puedan originarse entre los dos canales (*crosstalk*), éstos deben encontrarse aislados entre sí, de forma que cada canal cuente con su propia referencia a tierra flotante.

La importancia de aislar la señal de interés, además de para evitar el fenómeno *crosstalk* [9] anteriormente mencionado, radica en evitar que las posibles interferencias que puedan estar presentes a lo largo del circuito de acondicionamiento de la señal se introduzcan en la corriente que se desea generar a la salida del sistema. Y, es que, manejar corrientes de pequeña intensidad ($\pm$3mA) implica que cualquier pequeña variación suponga un cambio importante en la señal de interés, hecho que consideramos negativo e indeseado ya que se perdería el control de la funcionalidad del sistema.

Por otro lado, justo después de la ejecución de cada estímulo, el sistema llevará a ambos terminales de la carga conectada en cada canal a un potencial común, con el fin de eliminar la posible carga residual que quede almacenada en los electrodos y permitiendo, así, que cada canal pueda entregar y recuperar la corriente especificada, relentizando la degradación de los electrodos.

La información asociada a las formas de onda de corriente y los parámetros que definen el tipo de ejecución de

los estímulos estará contenida en ficheros elaborados por la persona usuaria manualmente o mediante las funciones ejecutadas en el software Matlab que han sido facilitadas (tanto la estructura de los ficheros como la información que deben contener serán explicadas en secciones futuras del presente documento). Esta información es la que será utilizada por el sistema software.

Dicho sistema constará de una tarjeta de desarrollo, en este caso del sistema Raspberry Pi, y de una PCB. El sistema Raspberry Pi se encargará de interpretar y enviar los datos contenidos en los ficheros definidos por la persona usuaria a la PCB, en donde serán acondicionados y transformados en los valores de corriente predefinidos inicialmente por la persona usuaria, creando las formas de onda que constituirán a los estímulos. Así mismo, el control y gestión de la ejecución de estímulos será llevado a cabo por el sistema Raspberry Pi.

La Figura 2-1 que se presenta a continuación representa un posible ejemplo de uso del sistema neuroestimulador, el cual, inyectará las formas de onda de corriente definidas previamente por la persona usuaria mediante Matlab sobre el sujeto en cuestión, a través de los dos canales aislados e independientes que dispone y gracias a los electrodos conectados en sus extremos. De este modo, los pines designados como '*Trigger I/O*' según el dibujo, permitirán lanzar o cortar la ejecución de los estímulos.

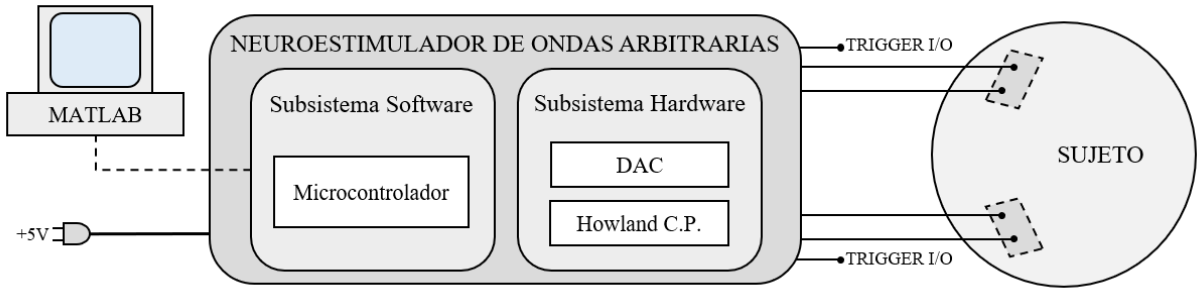


Figura 2-1. Caso de uso genérico del sistema neuroestimulador.

El sistema software ha sido diseñado para permitir tres modalidades de uso diferentes, designadas a lo largo de la descripción del proyecto como modos de ejecución o funcionamiento. Estos tres modos de ejecución se describirán en los siguientes apartados.

2.1.1 Primer modo de ejecución: *Trigger Out*

Este modo de ejecución da la posibilidad a la persona usuaria de definir el tiempo que existirá entre la ejecución de dos estímulos consecutivos (*inter stimuli time*). Este tiempo es el que se implementa en el control de las señales de disparo habilitadoras de los canales (*triggers*). La Figura 2-2 esboza los tiempos característicos de ejecución de una pila de estímulos para un canal cuando el dispositivo funciona según el primer modo de ejecución. Por simplificación, se ha representado únicamente la línea de tiempo de ejecución de dos estímulos.

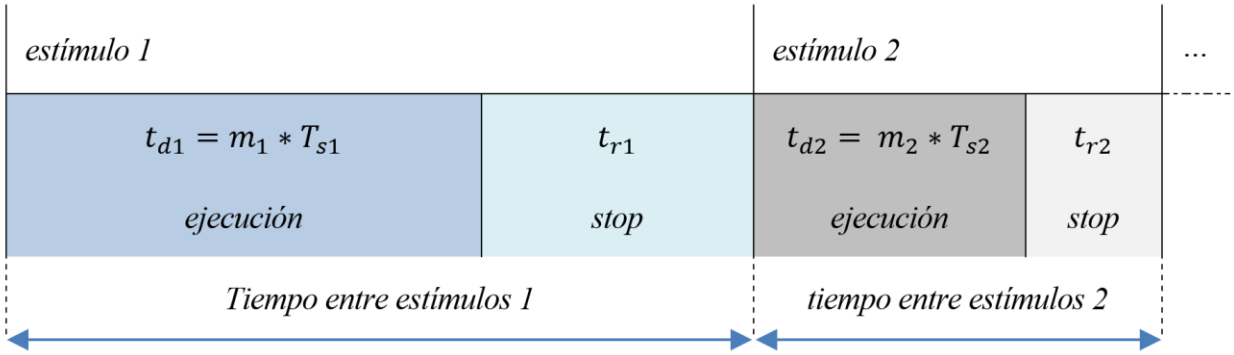


Figura 2-2. Línea de tiempo de la ejecución de una pila de estímulos según el primer modo de ejecución.

Los parámetros y variables que intervienen en la figura anterior quedan descritos en la Tabla 2-1,presentada a continuación:

Variable	Descripción
t_d	Tiempo de duración del estímulo
m	Número de muestras del estímulo
T_s	Periodo de muestreo del estímulo
t_r	Tiempo restante hasta el siguiente estímulo
t_{is}	Tiempo existente entre la ejecución de dos estímulos consecutivos

Tabla 2-1. Variables que caracterizan los tiempos característicos de ejecución de estímulos.

Este modo de ejecución permite la sincronización con sistemas externos, ya que, además de enviar hacia el exterior los estímulos de corriente, usa los pines designados como '*ETR_A*' y '*ETR_B*' de la PCB para enviar los pulsos de las señales de *trigger* que se implementan internamente para controlar la ejecución de estímulos. De esta forma, cualquier sistema dependiente de señales de habilitación, conectándose a estos pines, podrá sincronizarse con el dispositivo actual al recibir las mismas señales habilitadoras que se gestionan internamente en el neuroestimulador.

Para poder usar esta modalidad de ejecución, los *jumpers* equipados en la PCB deben estar conectados.

2.1.2 Segundo modo de ejecución: *Trigger In*

En este funcionamiento, a diferencia del primero, el control de las señales de *trigger* no queda predefinido, sino que es llevado a cabo mediante un dispositivo externo conectado a los pines '*ETR_A*' y '*ETR_B*' encargado de gestionar dichos pulsos habilitadores de ejecución.

En la siguiente Figura 2-3 se representan los tiempos característicos de ejecución de una pila de estímulos para un canal cuando el dispositivo funciona según el segundo modo de ejecución. Por simplificación, se ha representado únicamente la línea de tiempo de ejecución de dos estímulos.

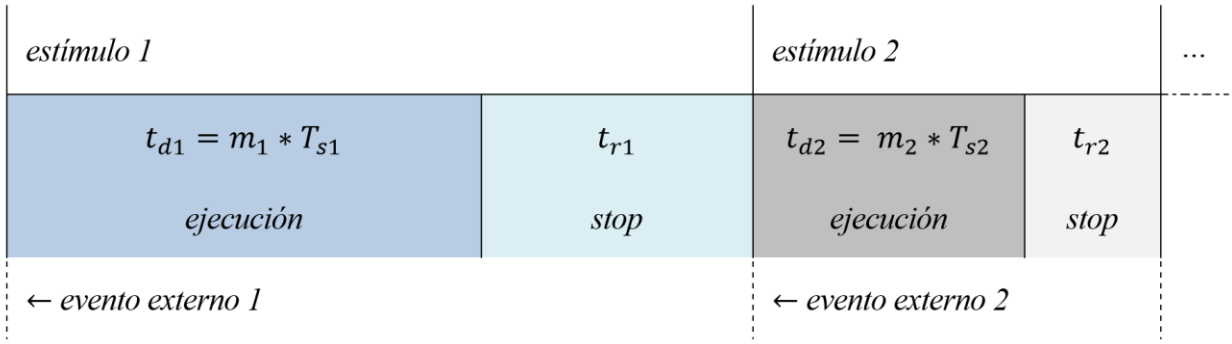


Figura 2-3. Línea de tiempo de la ejecución de una pila de estímulos según el segundo modo de ejecución.

Esta modalidad de funcionamiento permite sincronizar las formas de onda con el dispositivo externo encargado de gestionar las señales de *trigger*. Para este modo de ejecución, nuevamente es indispensable que los *jumpers* equipados en la PCB estén conectados.

Atendiendo a los dos modos de ejecución presentados, puede notarse cómo las pistas que conectan los pines '*ETR_A*' y '*ETR_B*' con el sistema Raspberry Pi son bidireccionales, ya que pueden adquirir un sentido de flujo de datos u otro dependiendo del modo de ejecución que se elija: el flujo de datos será en sentido desde el microcontrolador hacia el entorno para el primer método de ejecución y el flujo de datos será en sentido desde el entorno hacia el microcontrolador, para el segundo método de ejecución.

2.1.3 Tercer modo de ejecución: *Waves + Trigger*

El tercer y último modo de ejecución queda definido como '*Waves + Trigger*'. En este caso, el sistema neuroestimulador prescinde del control del sistema Raspberry Pi para lanzar y ejecutar los estímulos, pudiéndose encontrar apagada en todo momento de la ejecución. La persona usuaria, por tanto, podrá usar

dispositivos externos para generar tanto las formas de onda de corriente como las señales de habilitadoras de ejecución.

Mediante un generador de onda conectado en las entradas auxiliares del sistema '*IN_AUX_A*' e '*IN_AUX_B*' (para ello, en esta modalidad se deben retirar los *jumpers* provistos en la PCB), la persona usuaria podrá inyectar las señales, en valores de tensión, que desee convertir a corriente. Es importante que, a la hora de excitar, la persona usuaria tenga en cuenta la equivalencia voltaje – corriente del dispositivo, ya que el sistema implementa la conversión 1V → 1mA. De esta manera, por ejemplo, si la persona usuaria desea ejecutar pulsos de corriente de +2.6mA de amplitud, deberá configurar el generador de onda para que inyecte sobre el sistema pulsos de tensión de +2.6V de amplitud.

Así mismo, en este modo de ejecución, es posible controlar las señales de habilitación de estímulos mediante un dispositivo externo conectado a los pines '*ETR_A*' y '*ETR_B*' de la PCB.

Esta modalidad de funcionamiento convierte al sistema en un mero conversor de tensión a corriente a la vez que en un elemento de aislamiento de señales.

Estas diferentes posibilidades de uso permiten establecer un sistema software más genérico y flexible, orientado a poder usarse en un mayor número de aplicaciones destinadas a diferentes sectores, según el interés y las necesidades de la persona usuaria, e incluso integrando el dispositivo como subsistema neuroestimulador de otro conjunto de sistemas.

2.1.4 Requisitos funcionales

A continuación, en la Tabla 2-2 se presentan los límites de los requisitos funcionales del sistema neuroestimulador:

Requisitos funcionales	Mínimo	Máximo	Unidad
Frecuencia de muestreo	1	30k	Hz
Corriente de salida	-3	3	mA
Tiempo entre estímulos	0.5	60	s
Estímulos por canal	1	100	-

Tabla 2-2. Información y límites de los requisitos funcionales del sistema neuroestimulador.

2.2. Vista Física

El sistema neuroestimulador está controlado y sincronizado por la tarjeta de desarrollo Raspberry Pi. De este modo, es necesario disponer de un circuito de acondicionamiento de la señal encargado de procesar y transformar las señales digitales generadas por el microcontrolador, con el fin de conseguir a la salida del sistema los valores de corriente predefinidos por la persona usuaria. Así mismo, también es necesario un procesamiento de las señales de *trigger*.

La Figura 2-4 resenta un diagrama de bloques que esquematiza la arquitectura global del circuito de acondicionamiento. En el diagrama pueden distinguirse los subsistemas hardware que intervienen en el procesamiento de la señal y cómo quedan interconectados.

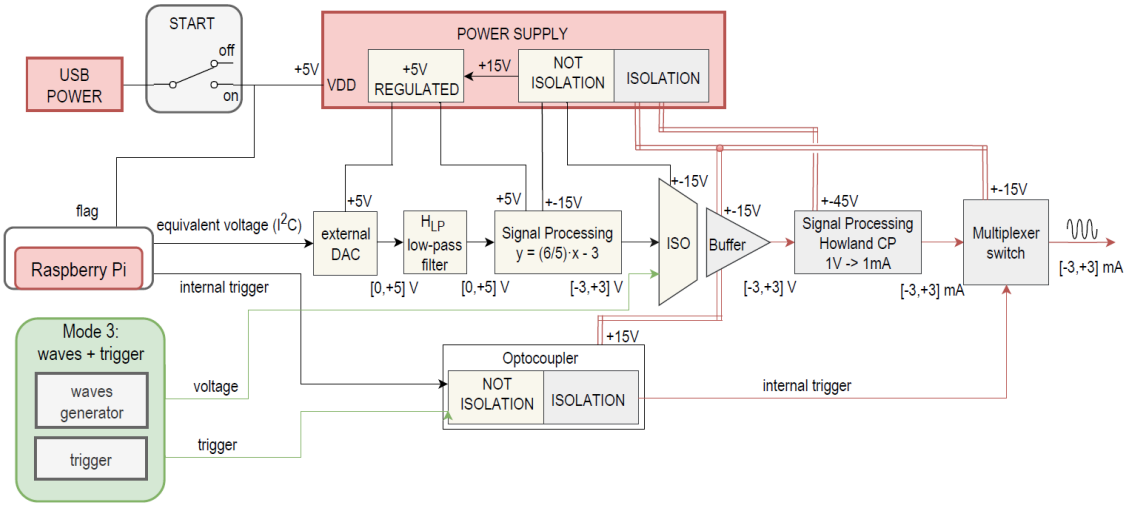


Figura 2-4. Diagrama de bloques que esquematiza la topología implementada del subsistema hardware del sistema neuroestimulador.

Atendiendo al formato mostrado en el diagrama de bloques, los bloques en color gris y las flechas y líneas en color rojo representan las etapas aisladas de la tierra común. El resto de conexiones y etapas no aisladas se han representado con bloques de color beige y flechas y líneas de color negro.

Un bloque de color verde con sus respectivas flechas verdes, ha sido empleado para indicar que, solo en el caso de que el sistema se encuentre en el tercer modo de ejecución '*Waves + Trigger*', la señal de excitación de entrada irá directamente sobre el amplificador de aislamiento, saltándose las etapas anteriores del circuito de procesamiento. La señal de *trigger*, generada de forma externa, al no ser controlada por el sistema Raspberry Pi, es inyectada directamente en el optoacoplador.

A continuación, se procederá a describir los subsistemas hardware que componen el sistema neuroestimulador.

2.2.1 Interruptor START

La adición de un interruptor deslizante permite cortar o suministrar la tensión general de alimentación a la PCB. Además, a través de un divisor resistivo, permite enviar un *flag* de aviso al microcontrolador, con objeto de indicar que el sistema dispone de tensión y se encuentra preparado para recibir, procesar y lanzar los estímulos.

Por otro lado, este interruptor permite abortar la operación en cualquier momento, ya que corta el suministro de voltaje en la PCB, notificando al sistema Raspberry Pi que se desea parar la ejecución de estímulos, momento a partir del cual, la Raspberry Pi detiene el envío de señales a la PCB.

Cabe destacar que la señal de aviso de ejecución por parte del interruptor al sistema Raspberry Pi solo tendrá efecto para el modelo de Raspberry Pi 2, ya que el valor de tensión de fábrica del pin al que se encuentra conectado es a nivel bajo, de forma que un incremento en el nivel de tensión del interruptor sería leído correctamente. No obstante, salvo por este detalle, el sistema neuroestimulador seguirá siendo igualmente funcional para el resto de modelos de Raspberry Pi.

2.2.2 Etapa de alimentación

A pesar de que la PCB está alimentada a +5V, existen diferentes etapas electrónicas que precisan de una alimentación dual muy superior y, además, dado que son etapas aisladas, la tensión de alimentación debe estar aislada también. El convertidor DC-DC de aislamiento de 1W de potencia '*TRACO POWER – TMA0515D*' ha sido empleado como elevador de tensión, ya que permite obtener, para una tensión de entrada de +5V, una tensión dual de salida de ±15V aislada de tierra, proporcionando una corriente de salida máxima igual a ±34mA (suficiente para nuestra causa). Así mismo, conectando tres convertidores en serie se consiguen los ±45V de alimentación que se requieren en la etapa de conversión voltaje – corriente.

Cada canal del sistema llevará su propia pila de tres convertidores debido a que deben estar aislados entre sí y con la tierra del sistema. De esta forma, los canales dispondrán de una conexión común a la tierra del sistema

(*GND*) en las etapas no aisladas. Para las etapas aisladas, cada canal dispondrá de su propia tierra virtual aislada. A su vez, ambos canales comparten un convertidor DC-DC con el fin de abastecer los ±15V no aislados que se requieran en determinadas etapas. Para obtener ±15V no aislados, bastará con cortocircuitar el pin '*COM*' del convertidor con la tierra común *GND*.

Por otro lado, determinadas etapas precisan puntos de tensión de referencia que deben ser precisos y estables, con un valor igual a +5V. Estos puntos de tensión, al no ser aislados, podría pensarse que para establecerlos se podría usar la tensión de alimentación global de la PCB, sin embargo, puesto que se necesita un valor de +5V exactos y que el diseño del sistema permite que la tensión de alimentación se encuentre alrededor de los +5V, se ha optado por usar el regulador de tensión '*ST – L7805CV*' con un encapsulado tipo *TO-220*. El regulador recibe en la entrada una tensión igual a +15V (no aislados) procedentes del convertidor DC-DC no aislado y proporciona +5V no aislados y regulados en su salida.

Finalmente, con el fin de reducir el posible rizado de la tensión de salida proporcionada por los convertidores y el regulador, se han incluido capacidades de 330nF (en los convertidores) y 100nF (en el regulador).

La siguiente tabla, Tabla 2-3, recoge los diferentes puntos de alimentación que se necesitan para abastecer al sistema neuroestimulador:

Cantidad	Componente	Tensión de salida	Dispositivo	Aislado
1	USB Tipo - A	+5V	PCB	No
1	USB Tipo - C	+5V	Raspberry Pi	No
1	TMA0515D	±15V	Canales 1 y 2	No
1	L7805CV	+5V Regulados	Canales 1 y 2	No
3 en serie	TMA0515D	±45V (también permite usar ±15V)	Canal 1	Sí
3 en serie	TMA0515D	±45V (también permite usar ±15V)	Canal 2	Sí

Tabla 2-3. Puntos de alimentación requeridos por el sistema neuroestimulador.

2.2.3 Etapa de conversion digital – analógico

En la primera etapa de la PCB, tiene lugar la conversión a tensión analógica de los datos digitales enviados por el sistema Raspberry Pi a través del protocolo síncrono de transmisión serie de datos I^2C. Para esta conversión se ha empleado el componente '*MICROCHIP – MCP4725*'. Este conversor digital a analógico (DAC) de 12 bits alimentado a +5V (aunque el rango permitido comprende desde los 2.7V a los 5.5V), presenta un tiempo de establecimiento de 6µs, el equivalente a una frecuencia de 166kHz, frecuencia muy superior a f_{MAX}=30kHz, la máxima requerida según los criterios de diseño mostrados en la Tabla 2-2, por lo que su uso es adecuado para la aplicación.

En el *pinout* de este elemento, puede distinguirse el pin '*A0*'. Este pin se corresponde con la selección de la dirección del dispositivo en lo que respecta a la comunicación I^2C. Para ello bastará con soldar este pin a la tensión de alimentación *VDD* (el dispositivo adquiere la dirección 0x61) o a la tensión de referencia *GND* (el dispositivo adquiere la dirección 0x60). Esta utilidad permitirá implementar dos DACs en un único bus I^2C.

Para estudiar la resolución que presenta el DAC, se puede obtener la diferencia ideal de voltaje entre dos códigos sucesivos o LSB como:

$$LSB_{ideal} = \frac{V_{REF}}{2^n} = \frac{5.0}{2^{12}} = \frac{5.0}{4095} = 1.22mV$$

Ecuación 2-1. Resolucion del DAC de 12 bits MCP4725.

La función que implementa el DAC puede expresarse según la Ecuación 2-2, en donde dac_{code} se corresponde con el valor comprendido entre 0 y 4095 que recibe del microcontrolador:

$$V_{out}^{DAC} = \frac{V_{REF}}{2^n} dac_{code} = \frac{5.0}{2^{12}} dac_{code}$$

Ecuación 2-2. Función de transferencia que implementa el DAC de 12 bits MCP4725.

Para establecer un modelo teórico y preciso del comportamiento del DAC y que se ajuste a la realidad con objeto de incluir este modelo en el software del sistema Raspberry Pi, es necesario que la tensión de referencia del convertidor sea lo más precisa posible. El regulador de tensión '*ST – L7805CV*' que se mencionó anteriormente, permite abastecer esta etapa con +5V regulados y constantes.

La gráfica de la izquierda de la Figura 2-5, muestra la característica estática del componente escogido. Tras alimentar el DAC a +5.00V se ha barrido el rango de funcionamiento permitido, de 0 a +5V en incrementos de +0.25V. Para ello, mediante la tarjeta de desarrollo Arduino conectada al DAC a través de los pines '*SDA*' y '*SCL*', vía puerto serie se le indica al Arduino el voltaje deseado que el DAC debe proporcionar. El Arduino transforma este voltaje en el código correspondiente (el número comprendido entre 0 y 4095) y lo envía al DAC. Una vez estabilizado el voltaje de salida en el DAC se ha medido con un multímetro digital (*DMM*) de la marca '*TACKLIFE – DM02A*'. Finalmente, vía Matlab, se ha representado la curva resultante. A la derecha de la Figura 2-5, se representa la precisión del DAC, graficando la diferencia entre el voltaje medido a la salida del DAC y el voltaje deseado que el DAC debe proporcionar.

Los resultados expuestos, denotan tanto la linealidad que caracteriza a este componente como la sensibilidad que presenta a la tensión de referencia *VDD*.

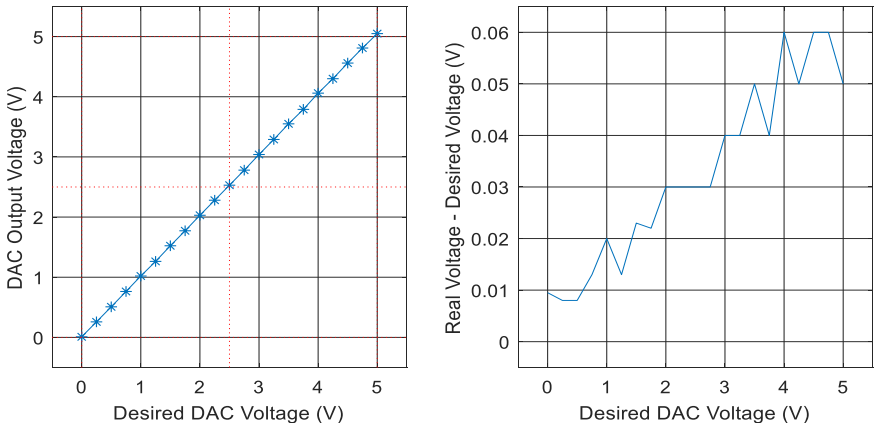


Figura 2-5. Característica estática del convertidor digital – analógico de 12 bits MCP4725.

2.2.4 Etapa de filtrado de paso de baja

Con el fin de suavizar el retenedor de orden cero (*ZOH*) para una mejor reconstrucción de la señal que proviene del DAC, se ha propuesto añadir un filtro de paso de baja a la salida del mismo. Para la construcción del filtro pasivo se ha decidido emplear una resistencia de valor 1kΩ y un condensador de valor igual a 1nF. El motivo de esta elección radica en situar la frecuencia de corte del filtro con los -3dB cercana a la frecuencia de trabajo del convertidor digital – analógico (mostrado anteriormente, ~166kHz) con el fin de mantener el mismo ancho de banda. El desarrollo teórico se representa en la Ecuación 2-3:

$$H_{LP}(s) = \frac{1/RC}{s + 1/RC} \rightarrow f_o = \frac{1}{2\pi RC} = 159.154\ kHz$$

Ecuación 2-3. Cálculo de la frecuencia de corte del filtro de paso de baja.

En la Figura 2-6, la gráfica de la izquierda expone la característica estática del filtro pasivo para un rango de entrada comprendido entre los 0 y +5V. La gráfica de la derecha recoge la caída de tensión con respecto a la tensión que proporciona el conversor digital – analógico, cuando la señal atraviesa dicho filtro.

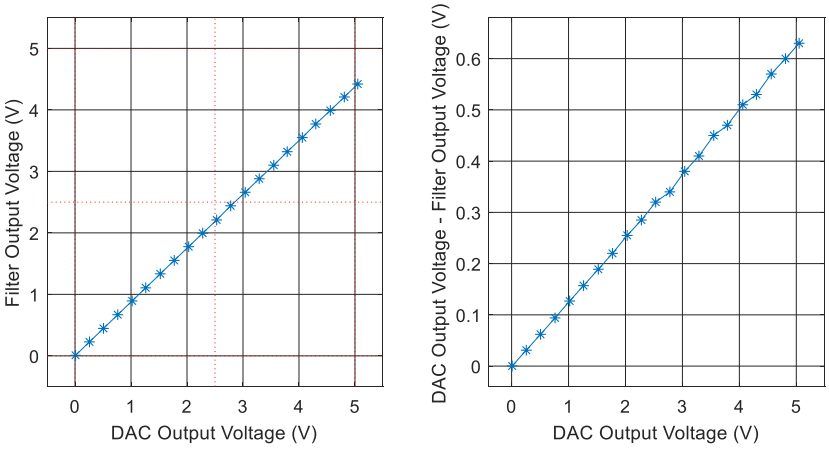


Figura 2-6. Característica estática del filtro paso de baja.

2.2.5 Etapa de ajuste de tension y amplificación

El DAC externo empleado proporciona un rango de tensiones comprendidas entre 0 y +5 V. Puesto que entre las especificaciones funcionales del sistema (Tabla 2-2) se proponen formas de onda arbitrarias y bifásicas, es decir, magnitudes tanto positivas como negativas, es necesario escalar la tensión proporcionada por el DAC en una tensión simétrica. En la etapa actual tiene lugar la implementación de dicha función de escalado.

Los límites simétricos de tensión que se proponen conseguir son ±3V, ya que en la siguiente etapa de procesamiento de la señal se realizará la conversión voltaje – corriente con una ganancia igual a 1e-3, es decir, se buscará la equivalencia de 1V → 1mA. De esta forma, dado que las especificaciones iniciales proponen un sistema capaz de entregar ±3mA de corriente, se necesitarán, por tanto, hacer llegar a esta etapa de conversión voltaje – corriente, ±3V.

En la Ecuación 2-4, se muestra una sencilla función matemática que implementa el escalado de voltaje de forma lineal de [0, +5] V a [-3, +3] V:

$$f(x) = \frac{6}{5}x - 3$$

Ecuación 2-4. Expresión matemática que implementa el escalado de tensión necesario para el sistema neuroestimulador.

En la implementación electrónica de la función de transferencia anterior se propone hacer uso de un amplificador operacional (*opamp*) en configuración de sumador diferencial. El componente empleado ha sido el '*TEXAS INSTRUMENTS – TL082CP*' con el encapsulado *DIP8*, debido a que, según la hoja de datos del fabricante, presenta un alto ancho de banda (4MHz) y un bajo consumo de corriente de alimentación. La topología de sumador diferencial requiere de un *opamp* con una alta impedancia de entrada, con el fin de que la caída de tensión entre los terminales de entrada del amplificador sea la mínima posible. El componente *TL082CP* cuenta con una alta impedancia de entrada igual a $10^{12}\Omega$, por lo que es adecuado su uso.

La topología final implementada en la etapa de ajuste de tensión queda expuesta en la Figura 2-7 que se presenta a continuación:

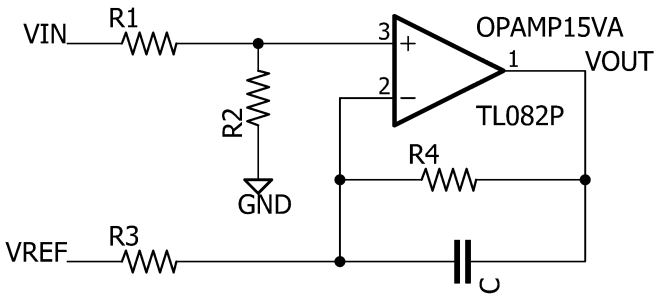


Figura 2-7. Topología de la etapa de ajuste de tensión del subsistema hardware del sistema neuroestimulador.

Con el fin de determinar los valores de las resistencias y del punto de voltaje referencia V_{REF}, se ha realizado un breve análisis del circuito anterior mediante ecuaciones nodales y asumiendo el *opamp* como componente ideal (la diferencia de tensión entre los terminales de entrada del amplificador es igual a cero y la impedancia de entrada del mismo es infinita, por lo que la corriente de entrada entre sus terminales es cero), obteniendo la siguiente ecuación que relaciona la entrada y la salida de la etapa actual.

$$V_{out} = \frac{1}{R_3}\left(\left(Vin\frac{R_2}{R_1+R_2}\right)(R_3+R_4) - R_4V_{REF}\right)$$

Ecuación 2-5. Análisis nodal de la etapa de ajuste de tensión del subsistema hardware del sistema neuroestimulador.

Para que la Ecuación 2-5 cumpla la función de transferencia que implementa la Ecuación 2-4, los valores de los parámetros que la conforman deben ser:

$R_1 = 2k\Omega$ $R_2 = 6k\Omega$ $R_3 = 5k\Omega$ $R_4 = 3k\Omega$ $V_{REF} = 5.0V$

Una consideración a tener en cuenta es que, en las etapas que preceden a la actual (el conversor digital a analógico seguido del filtro de paso de baja), tiene lugar una leve caída de tensión con respecto a la original producida por el propio DAC, como consecuencia de la resistencia de 1kΩ usada en el filtro pasivo. Este hecho es el responsable de que se deba reducir el valor de la resistencia $R_1 = 2k\Omega$ a un valor igual a 1kΩ (estas dos resistencias en serie ya sumarían los 2kΩ que se exigen en el diseño de la etapa actual). Además, con el fin de filtrar las posibles espigas que puedan originarse a la salida del amplificador operacional, se ha incluido una capacidad de 1pF en paralelo con la resistencia R_4, la cual no modifica considerablemente el ancho de banda de la etapa actual, pero incrementa la estabilidad de la misma.

Cabe destacar que los +5.00V requeridos en esta topología son suministrados a través del regulador de tensión *L7805*, tal y como se comentó anteriormente en el apartado: Etapa de alimentación.

Mediante el software de simulación electrónica LTSpice y representando mediante figuras en Matlab, en las dos gráficas de la izquierda de la Figura 2-8 se ha representado el voltaje de salida de la etapa actual frente al voltaje generado por el DAC (se ha modelado el DAC como una fuente de tensión constante). Recordando que el objetivo de esta etapa es el de escalar la tensión que proviene del DAC entre los límites de ±3V, puede comprobarse que la topología implementada cumple con los requisitos de diseño. Además, se ha realizado un análisis AC, vía LTSpice, para estudiar la respuesta frecuencial del conjunto filtro de paso de baja y la actual etapa de escalado de voltaje. Así mismo, en la gráfica derecha de la Figura 2-8, puede comprobarse que el resultado de la simulación aproxima al conjunto con un ancho de banda alrededor de los 180kHz (frecuencia correspondiente a la caída de -3dB de la ganancia). No obstante, este ancho de banda quedará limitado al ancho de banda del convertidor digital – analógico (~166kHz).

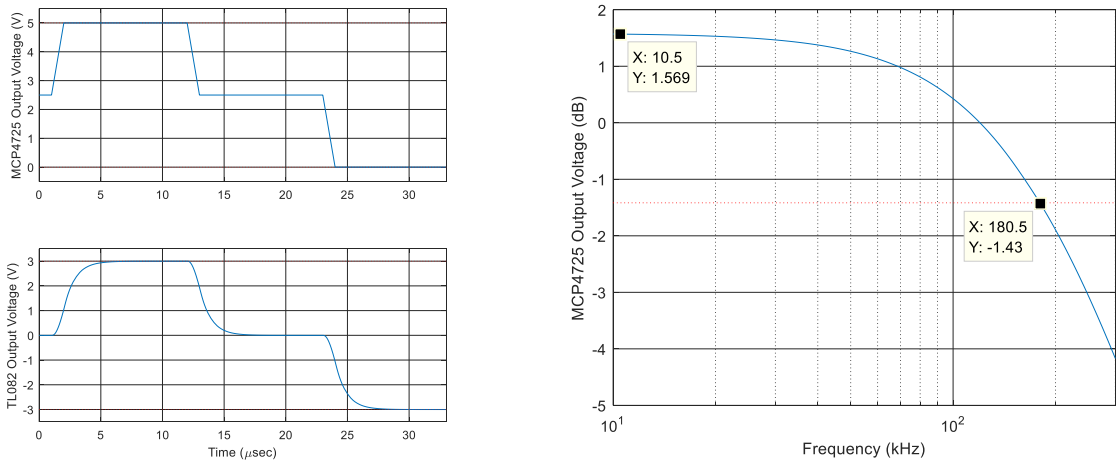


Figura 2-8. Resultados de la simulación transitoria en LTSpice de la etapa de ajuste de tensión del subsistema hardware del sistema neuroestimulador.

2.2.6 Etapa de aislamiento de la señal

Con objeto de conseguir el aislamiento por completo de los canales del sistema, en el punto actual tiene lugar el aislamiento de las señales correspondientes a los estímulos.

El componente '*TEXAS INSTRUMENTS – ISO124-P*' ha sido empleado para llevar a cabo el aislamiento de la señal, gracias a que, entre otras cosas, presenta ganancia unidad (no altera las señales de tensión que recibe, por lo que lo definiremos como *buffer* de aislamiento) y puede alimentarse hasta una tensión máxima de ±18V, de forma que se usarán las salidas de ±15V de los convertidores DC-DC, tanto aisladas como no aisladas, tal y como se indican en las especificaciones del componente. A su vez, este componente resulta de interés, ya que cuenta con un consumo de corriente de alimentación reducido, que, según las especificaciones del fabricante es de hasta un máximo de ±5mA por cada alimentación: la aislada y la no aislada.

En esta etapa de aislamiento se encuentra el principal factor limitante del ancho de banda del sistema. Anteriormente, queda visto que el ancho de banda que presentan las etapas ya estudiadas se correspondía con el ancho de banda del convertidor digital – analógico, ~166kHz, sin embargo, la adición de la actual etapa de aislamiento limita el ancho de banda del sistema a 50kHz, ancho de banda indicado en las especificaciones del amplificador de aislamiento empleado (*ISO124-P*). A pesar de esta considerable reducción, 50kHz siguen estando dentro del rango de frecuencias permitido según los requisitos iniciales de diseño (generar formas de onda arbitrarias de hasta un máximo de 30kHz).

Con vistas a dar la posibilidad de operar en el tercer modo de ejecución, explicado anteriormente en el apartado Vista Lógica, se ha incluido un *jumper* entre la etapa anterior (etapa de ajuste de tensión) y la etapa actual (*buffer* de aislamiento). Esta sencilla modificación permite inyectar señal directamente sobre el *buffer* si la señal de excitación se conecta directamente al pin del *jumper* designado en la PCB como '*IN_AUX*', permitiendo así, operar según el tercer modo de ejecución. Para poder operar en los otros dos modos, bastará con cortocircuitar ambos pines del *jumper*, de manera que la salida del operacional *TL082* quedará conectada a la entrada del buffer de aislamiento *ISO124-P*.

En la Figura 2-9, la gráfica de la izquierda presenta el estudio de la característica estática del amplificador de aislamiento, el cual se ha alimentado a ±15V, tanto aislados como no aislados. Excitando con tensiones comprendidas entre los -3V y +3V que recibirá en la práctica y midiendo la salida del *buffer* con el multímetro digital que se ha estado empleando en los experimentos anteriores, los resultados obtenidos verifican la ganancia unidad que presenta el amplificador. No obstante, la gráfica de la derecha profundiza en la precisión del dispositivo al mostrar la diferencia de tensión entre la salida y la entrada del *buffer*.

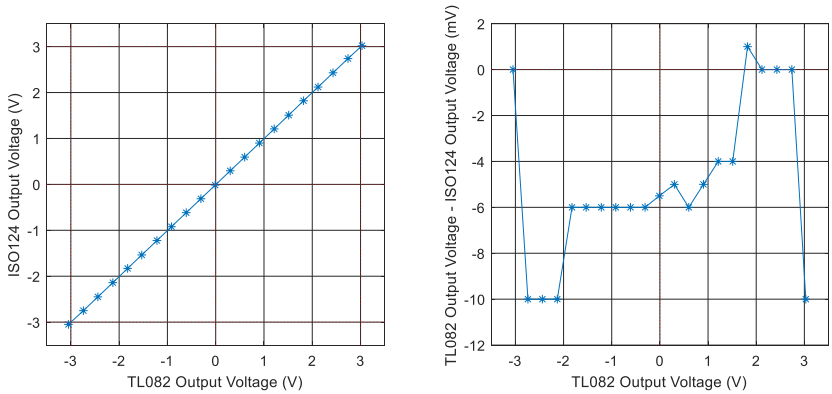


Figura 2-9. Característica estática del amplificador de aislamiento ISO124-P del subsitema hardware del sistema neuroestimulador.

2.2.7 Etapa de conversion tension – corriente: Fuente de corriente Howland

En la actual y penúltima etapa de procesamiento de la señal, tiene lugar la conversión a corriente del voltaje aislado que se recibe procedente del *buffer* de aislamiento. Los valores de corriente resultantes, para el caso del primer y segundo modo de ejecución, deberán coincidir con los valores de corriente predefinidos inicialmente por la persona usuaria.

Entre las diferentes topologías de fuentes de corriente que pueden encontrarse, se ha decidido emplear la fuente de corriente Howland por, entre otros motivos, el buen desempeño que presenta, por hacer uso de un

único amplificador operacional en su topología y por proporcionar una salida bipolar, lo que le permite operar en dos cuadrantes al proporcionar corriente tanto positiva como negativa [10].

Para la construcción de la fuente de corriente Howland, se partirá inicialmente de la topología de amplificador diferencial, ya que este circuito constituye la base sobre la que se diseña la fuente de corriente. Por ello, para comprender tanto el diseño como el funcionamiento de la fuente de corriente, se tomará inicialmente un amplificador diferencial ajustado para obtener ganancia unidad. Véase la topología en la Figura 2-10:

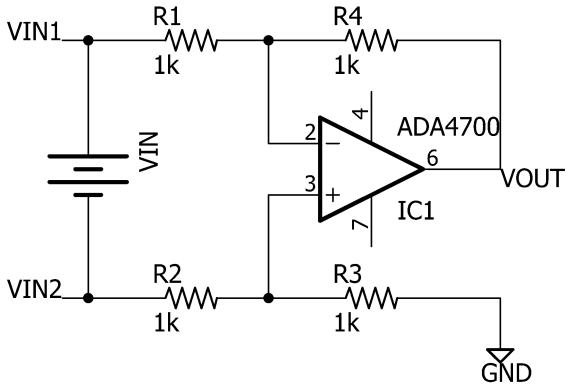


Figura 2-10. Topología del amplificador diferencial configurado con ganancia unidad.

El amplificador diferencial se caracteriza por proporcionar una tensión de salida que depende únicamente de la diferencia de tensión entre los terminales de entrada (*VIN2–VIN1*).

Esta propiedad queda demostrada en la gráfica expuesta en la Figura 2-11, la cual incluye los resultados tras simular en LTSpice la topología de amplificador diferencial anterior, para el caso en el que la referencia a tierra (*GND*) se encuentre en el nodo *VIN1* y para el caso de situar la referencia a tierra (*GND*) en el nodo *VIN2*. En ambas simulaciones, la fuente de tensión de entrada *VIN* es la misma.

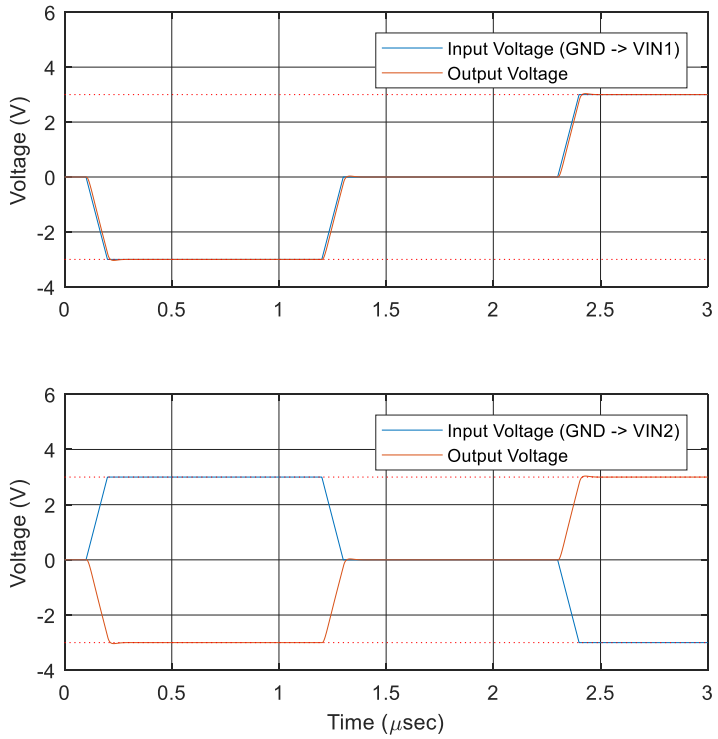


Figura 2-11. Resultados de la simulación en LTSpice del amplificador diferencial configurado con ganancia unidad.

Con los resultados expuestos, puede comprobarse que, independientemente de donde se coloque la referencia a tierra, el voltaje de salida no cambia, ya que como se mencionó anteriormente, el voltaje de salida de la topología solo depende de la diferencia de tensión entre los terminales de entrada.

Seguidamente, el siguiente paso en la construcción de la fuente de corriente Howland consiste en añadir una resistencia, la cual designaremos como R_{3B}, entre el terminal de salida del operacional y tierra. Además, se eliminarán las referencias a tierra de las ramas de entrada del circuito, tal y como se ha probado en el experimento anterior. De esta forma, de esta forma, todo el circuito quedará referenciado a la tierra situada entre las resistencias R_{3A} y R_{3B} (nótese como R_3 ha pasado a denominarse R_{3A}).

El efecto que produce la variación del valor de R_{3B} sobre la funcionalidad de la topología anterior queda plasmado en la gráfica de la Figura 2-12. Simulando nuevamente con LTSpice y realizando las gráficas vía Matlab, se presentan los resultados cuando R_{3B} vale 100Ω y para cuando vale 10kΩ. En ambos casos la tensión de entrada *VIN* es constante e igual a +1V (nótese que *VIN2–VIN1*=-1V).

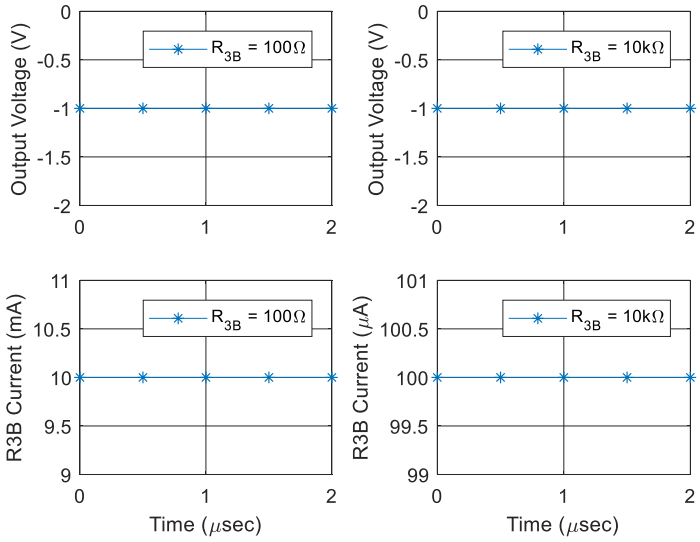


Figura 2-12. Resultados de la simulación en LTSpice del amplificador diferencial configurado con ganancia unidad y modificado con la adición de la resistencia R_{3B} y ante variaciones del valor de la misma.

Comparando los resultados anteriores, se comprueba que añadir la resistencia R_{3B} no afecta en la funcionalidad del circuito, ya que la ganancia del sistema sigue siendo igual a 1. No obstante, gracias a esta resistencia adicional, se puede conducir diferentes valores de corriente a través de ella sin alterar la tensión de salida *opamp*. Nótese, además, que se puede predecir el valor de la corriente de interés puesto que, al usar una topología de amplificador diferencial configurado como *buffer*, conociendo el valor de la resistencia R_{3B}, el valor de la corriente es fácilmente calculable empleando la '*Ley de Ohm*'. Véase la siguiente ecuación, Ecuación 2-6:

$$V = I \cdot R \rightarrow I_{R_{3B}} = -V_{IN}/R_{3B}$$

Ecuación 2-6. Cálculo de la corriente que circula por la resistencia R_{3B} de la topología de amplificador diferencial modificado y configurado con ganancia unidad, empleando la Ley de Ohm.

Para continuar con el estudio, con objeto de aislar la resistencia que se ha añadido para forzar que toda la corriente circule a través de ella y evitando posibles pérdidas de corriente [11], se propone convertir la tierra de la resistencia R_{3A} en tierra virtual o flotante haciendo uso de un *buffer*. La resistencia R_{3B} será de valor igual a 1kΩ para conseguir la equivalencia 1V → 1mA. El circuito resultante se presenta en la Figura 2-13:

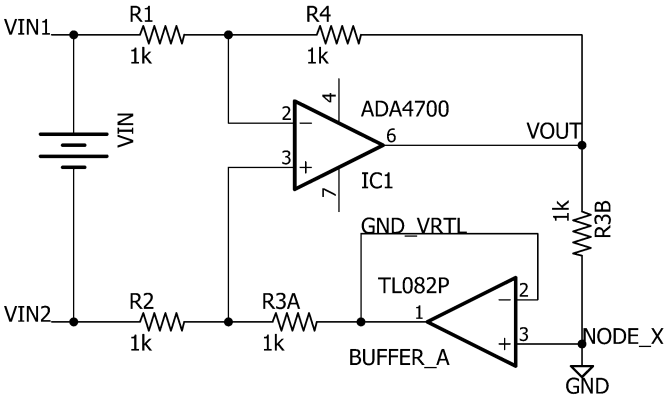


Figura 2-13. Topología del amplificador diferencial configurado con ganancia unidad y modificado con un buffer para conseguir el aislamiento de la R_{3B} con respecto a GND.

La adición del amplificador en configuración de *buffer*, idealmente, permite el aislamiento por completo de la resistencia R_{3B}. La elevada impedancia de entrada del amplificador operacional (idealmente infinita) evita que una porción de la corriente que atraviesa la resistencia R_{3B} se divida hacia el *buffer*, permitiendo que toda la corriente circule hacia el único punto de tierra absoluta (*GND*) que hay en la topología. Del mismo modo, el *buffer* permite colocar la tensión de entrada (en este caso, referencia a tierra) en la salida, ya que cuenta con una impedancia de salida muy baja (idealmente igual a cero). De esta forma, el *buffer* permite establecer una referencia a tierra (virtual) en el nodo definido como *GND_VRTL*, respetando, así, la topología inicial.

La Figura 2-14 incluye los resultados de un sencillo experimento, ideado con el fin de estudiar el efecto que produce en la topología alterar la tensión del nodo *X,* designado en la Figura 2-13. De este modo, para la simulación se ha colocado una fuente de tensión (de +5V, por ejemplo) en este nodo y se han representado las tensiones de los nodos de interés y la corriente que circula por la resistencia R_{3B}.

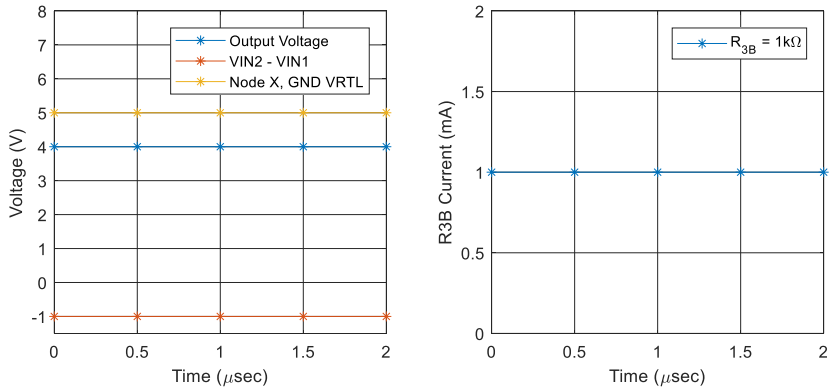


Figura 2-14. Resultados de la simulación en LTSpice del amplificador diferencial configurado con ganancia unidad y modificado con un buffer para conseguir el aislamiento de la R_{3B} con respecto a GND cuando el nodo X se fuerza a una tensión de +5V.

Como se mencionó anteriormente, la tensión de salida de la topología implementada depende únicamente de la ganancia de la configuración y de la diferencia de tensión entre los terminales de entrada. Observando los resultados, se aprecia que es independiente del voltaje que presente el nodo *X*. Puede comprobarse cómo coincide el valor de la tensión de entrada y el de salida en los resultados del experimento de la Figura 2-14, la diferencia de tensión entre los terminales de entrada del amplificador *VIN2–VIN1* es igual a -1V y la tensión de salida (calculada como $VOUT–V_{NODE\_X} = 4V – 5V$), es igual a -1V.

El objetivo de estas pruebas es el de comprobar que modificar la tensión del nodo *X* escala la tensión del resto del circuito, aunque la función de transferencia que implementa se ve inalterable. De esta forma, se demuestra que con la topología actual se consigue una tensión constante a la salida del *opamp* , lo que se traduce en una corriente constante sobre la resistencia R_{3B}, argumento clave y necesario para realizar el último paso de la construcción de la fuente de corriente Howland.

Así, finalmente, el último paso para construir la fuente de corriente Howland consistirá en añadir la carga sobre la que se desea inyectar la corriente constante. En otras palabras, se añadirá una resistencia entre el terminal positivo de entrada del *buffer* y la referencia a tierra, con el fin de modelar la carga que se colocará en

un futuro: los electrodos. Por consiguiente, se consigue inyectar una corriente controlada a través de la resistencia de carga, la cual puede valer lo que se desee dentro de un rango acotado por un valor máximo determinado por el voltaje que cae sobre esta resistencia; voltaje que debe ser capaz de generar el *opamp*.

Con el propósito de construir una fuente de corriente con un solo amplificador operacional, según lo mencionado al comienzo de la sección actual, se eliminará el amplificador configurado como *buffer* de la topología en cuestión, quedando así unidas las resistencias R_{3A} y R_{3B} a la tierra común. Esta modificación supone una alteración del valor de la corriente que ve la resistencia de carga, como consecuencia de la creación de un camino alternativo por la rama no inversora del amplificador restante, por el que circula parte de la corriente determinada por la resistencia R_{3B}.

Para evitar este fenómeno, se incrementará el valor de la impedancia de esta rama no inversora con respecto a la impedancia de la resistencia R_{3B}, ya que así, se consigue que la corriente, casi en su totalidad, circule hacia la resistencia de carga al disponer de un camino que presenta menor resistencia al paso de corriente.

El circuito resultante, denominado fuente de corriente Howland mejorada [12], es el mostrado a continuación en la Figura 2-15:

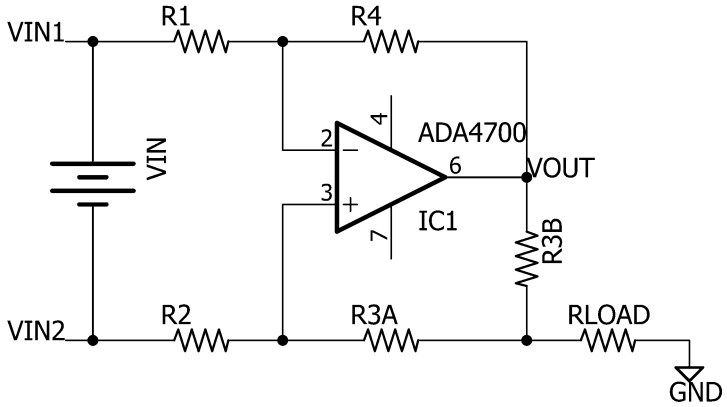


Figura 2-15. Topología de la fuente de corriente Howland mejorada con una resistencia en la salida que modela una posible carga.

Para la elección de los valores de las resistencias, se deben tener en cuenta una serie de aspectos que determinarán, entre otros, el rendimiento de la fuente de corriente Howland mejorada:

a) Realizando un breve análisis nodal de la fuente de corriente Howland mejorada, suponiendo un amplificador operacional ideal, se puede expresar la corriente que circula por la carga como se muestra en la Ecuación 2-7:

$$I_L = -V_{IN}\frac{R_4(R_2 + R_{3A})}{R_1R_{3B}(R_2 + R_{3A}) + R_L(R_1R_{3A} + R_1R_{3B} - R_2R_4)}$$

Ecuación 2-7. Análisis nodal de la corriente que circula por la resistencia de carga colocada en la salida de la fuente de corriente Howland mejorada.

Si, para la fórmula anterior, se suponen $R_1 = R_4$ y $R_2 = R_{3A} + R_{3B}$, la corriente que circula por la resistencia de carga, puede expresarse nuevamente según la Ecuación 2-8:

$$I_L = -\frac{1}{R_{3B}}V_{IN}$$

Ecuación 2-8. Análisis nodal simplificado de la corriente que circula por la resistencia de carga colocada en la salida de la fuente de corriente Howland mejorada.

La sencilla Ecuación 2-8, invita a pensar que, para mantener de una forma precisa la equivalencia voltaje – corriente , tal que 1V a la entrada de la fuente de corriente proporcione 1mA en la resistencia de carga, ambas ramas del circuito deben encontrarse perfectamente balanceadas, de forma que las resistencias que intervienen se ajusten a ciertos ratios que pueden expresarse como:

$$\frac{R_1}{R_2} = \frac{R_4}{R_{3A} + R_{3B}} = 1$$

Ecuación 2-9. Ratios que deben cumplir las resistencias que conforman la topología de la fuente de corriente Howland mejorada.

De esta manera, puede comprobarse la influencia que ejercen las tolerancias de fabricación de las resistencias [12] sobre el rendimiento ideal de la fuente de corriente Howland. Por ello, uno de los criterios de selección de las resistencias a prestar una especial atención es elegir resistencias con la menor tolerancia posible.

En la Figura 2-16 se incluyen demostraciones del efecto que provoca disponer de una topología con ramas no balanceadas. Como en casos anteriores, para la simulación se ha usado una fuente de tensión en la entrada de la etapa actual igual a +1V.

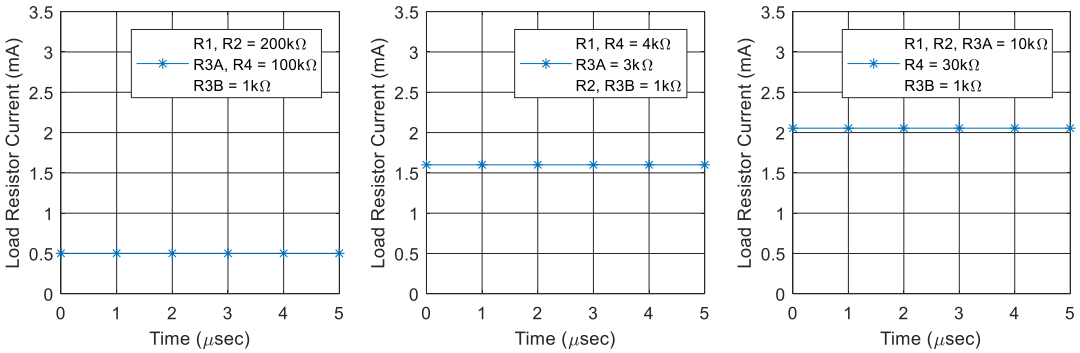


Figura 2-16. Resultados de la simulación transitoria en LTSpice de la corriente que circula por la resistencia de carga de la fuente de corriente Howland mejorada ante diferentes valores de las resistencias de la topología.

b) Valores grandes de las resistencias de los lazos de realimentación del amplificador pueden provocar un incremento de la corriente de polarización en los terminales de entrada del amplificador, de forma que se puede crear una tensión de *offset* indeseada [13]. Del mismo modo, aumentar los valores de las resistencias, aunque disminuya el consumo de corriente de alimentación, disminuye el ancho de banda de la etapa actual e incrementan el ruido térmico. El análisis frecuencial mostrado en la Figura 2-17 detalla que, además de reducirse el ancho de banda al incrementar el valor de las resistencias, aparece un pico de resonancia cuya magnitud no debe pasarse por alto, ya que ~13dB equivalen a más de 4V.

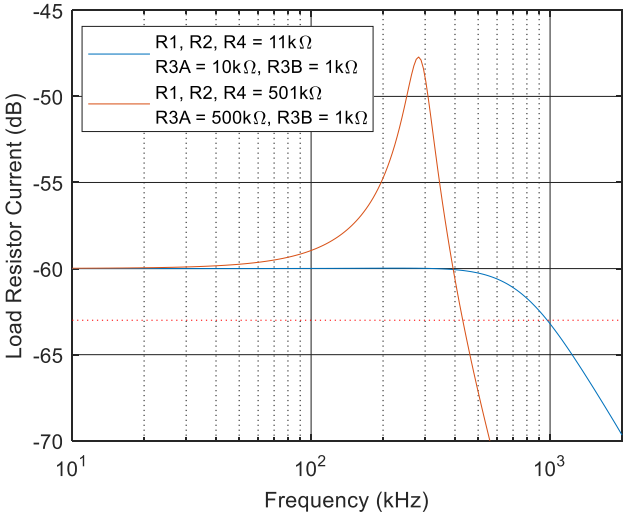


Figura 2-17. Resultados de la simulación frecuencial en LTSpice de la corriente que circula por la resistencia de carga de la fuente de corriente Howland mejorada ante diferentes valores de las resistencias de la topología.

Atendiendo a los puntos anteriores, se han escogido resistencias con tolerancias de fabricación igual a un 1%, y con los valores que se presentan a continuación:

$R_1 = 51k\Omega$ $R_2 = 51k\Omega$ $R_{3A} = 50k\Omega$ $R_{3B} = 1k\Omega$ $R_4 = 51k\Omega$

La implementación física de las resistencias que valen 51kΩ se llevará a cabo usando dos resistencias en serie: una de 50kΩ y otra de 1kΩ.

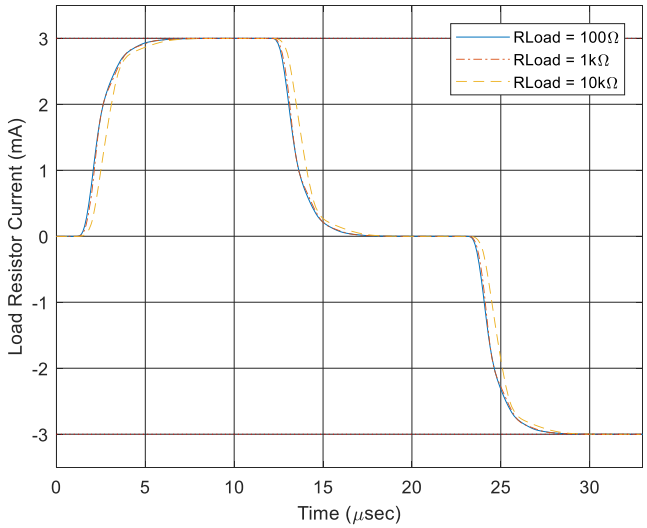
Cabe anotar que en el diseño de una fuente de corriente Howland una de las mayores limitaciones, junto a las que ya se han comentado, es la correcta selección del amplificador operacional con que se implementará. La ganancia de lazo abierto del amplificador operacional y el ancho de banda, constituyen unos de los principales parámetros a tener en cuenta [10]. Sin embargo, en el actual proyecto, el criterio de selección de mayor peso lo adquiere la tensión que es capaz de soportar y generar el amplificador operacional. Fijándonos en las especificaciones de diseño del proyecto, se requiere de un dispositivo capaz de entregar formas de onda de corriente de hasta ±3mA. Para esta corriente, emplear cargas de 15kΩ, por ejemplo, implica que el operacional suministre como mínimo los ±45V que caerán sobre la carga.

El amplificador elegido ha sido el '*Analog Devices – ADA4700-1*', un operacional capaz de funcionar con tensiones de hasta ±50V y con un ancho de banda igual a 3.5MHz.

Por otro lado, se ha visto conveniente añadir dos condensadores de valor igual a 1pF, cada uno en paralelo con las resistencias R_{3A} y R_{3B}, con el fin de evitar las posibles espigas de tensión ocasionadas por la conmutación que tiene lugar en la siguiente etapa (explicado en el punto Interruptor digital: Multiplexor). Además, se propone añadir una red de compensación '*lead – lag*' [12], es decir, incluir entre los terminales de entrada del amplificador una resistencia de valor igual a 2kΩ en serie con un condensador de valor igual a 1.8pF. Esta red de compensación tiene por objeto mejorar la estabilidad del sistema sin degradar el ancho de banda de la impedancia de salida.

Por último, dado que la etapa actual es sensible a la diferencia de tensión entre sus terminales de entrada y que la señal de interés, referenciada a la tierra flotante del canal, es inyectada en uno de los terminales de entrada, el otro terminal de entrada deberá ir conectado a esa misma tierra flotante, con el fin de mantener la misma referencia de tensión.

Para verificar el diseño del sistema y comprender el efecto que produce incrementar el valor de la resistencia de carga, se han simulado con LTSpice las respuestas transitorias y frecuenciales del conjunto de etapas compuesto por el filtro paso de baja, la etapa de escalado de tensión y la fuente de corriente Howland mejorada. Como señal de entrada, se ha modelado nuevamente el convertidor digital – analógico como una fuente de tensión constante, la cual barre los valores límites de tensión que el DAC puede aportar: +5V, +2.5V y 0V. Los resultados son expuestos a continuación, en la Figura 2-18:



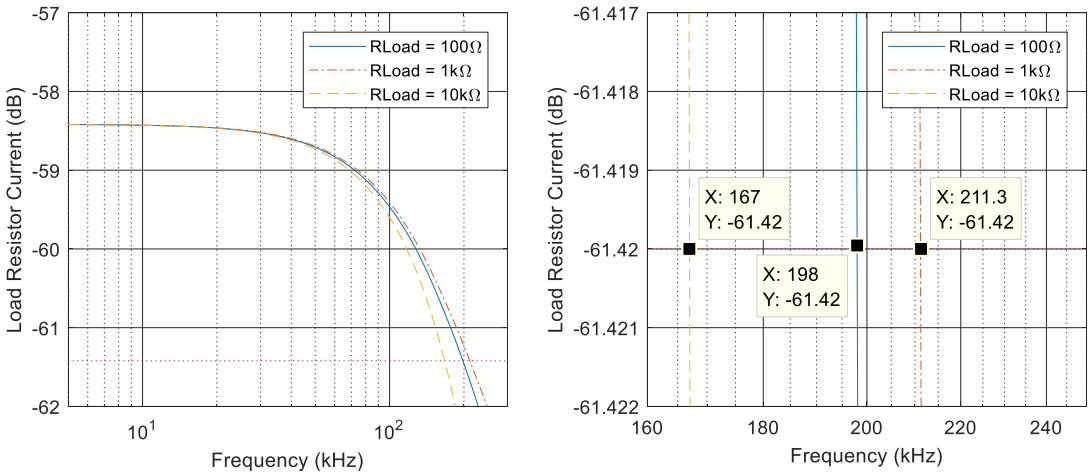


Figura 2-18. Resultados de la simulación transitoria y frecuencial en LTSpice de la corriente que circula por la resistencia de carga conectada al sistema compuesto por una fuente de tensión constante que modela el DAC, el filtro paso de baja, la etapa de ajuste de tensión y la fuente de corriente Howland mejorada ante variaciones en la resistencia de carga de la salida.

En la gráfica superior de la Figura 2-18 se ha representado la respuesta temporal de la corriente que atraviesa la resistencia de carga. Así mismo, para facilitar la comprensión de los resultados, se han realizado las gráficas para los valores de la corriente que el sistema debe proporcionar como consecuencia de la excitación de entrada usada. Cabe recordar, que estos valores se corresponden, además, con el límite máximo y mínimo de corriente que el sistema debe ser capaz de alcanzar.

En la gráfica inferior de la Figura 2-18, se ha representado la respuesta frecuencial de la corriente que atraviesa la resistencia de carga. De igual manera, se ha trazado una recta horizontal que representa la magnitud de corte con los -3dB. Con el fin de esclarecer los resultados, también se han representado los resultados tras realizar un *zoom* en la zona de interés de la gráfica central.

Los resultados anteriores explican que diferentes valores de la resistencia de carga implican valores diferentes en el ancho de banda del sistema. No obstante, aunque el tiempo de respuesta del sistema se vea afectado, se comprueba fácilmente que el amplificador de aislamiento usado (descrito en la sección anterior) sigue siendo el elemento más restrictivo en lo que respecta al ancho de banda, ya que como se mencionó previamente, contaba con una respuesta frecuencial constante hasta los 50kHz, y es que, en la gráfica anterior, se comprueba que el ancho de banda del sistema es superior a los 160kHz (recordando que las simulaciones no incluyen el *buffer* de aislamiento mencionado).

Para completar el estudio teórico, se ha propuesto realizar un análisis transitorio y frecuencial para el caso en el que el amplificador usado en la fuente de corriente Howland mejorada no pueda abastecer a lo que se le pide, con el fin de analizar los resultados y estimar el comportamiento que presentará en la práctica. Para el ejemplo, se ha simulado el sistema empleando una resistencia de carga de valor igual a 25kΩ.

Los cálculos previos indican que, para una corriente esperada de ±3mA, la tensión que deberá proporcionar el amplificador será, como mínimo, igual a ±75V. Recordando que el amplificador usado se encuentra alimentado a ±45V, es fácil predecir que el sistema no será capaz de entregar la corriente requerida en las especificaciones iniciales.

Para comprobar lo anterior, los resultados de las simulaciones se presentan en la Figura 2-19:

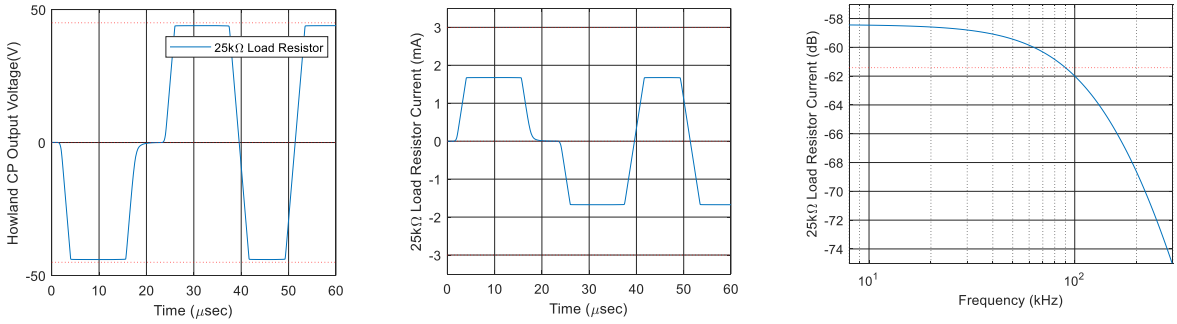


Figura 2-19. Resultados de la simulación transitoria y frecuencial en LTSpice de la corriente que circula por la resistencia de carga igual a 25kΩ conectada a la salida de la fuente de corriente Howland mejorada, con el fin de observar el comportamiento de la propia fuente de corriente ante exigencias superiores a las de los requisitos de diseño del sistema neuroestimulador.

La gráfica izquierda de la imagen anterior representa el comportamiento temporal del voltaje de salida del amplificador usado en la fuente de corriente. Se han incluido dos rectas horizontales que representan la tensión de alimentación, ±45V. En esta gráfica, puede verse la saturación que presenta la señal cercana a los límites de alimentación cuando se excita al sistema para obtener los ±3mA. Esta limitación de tensión se traduce en un recorte de la corriente, tal y cómo se muestra en la gráfica central, en la que se representa la corriente que circula a través de la resistencia de carga. El valor de la corriente nunca alcanza el valor exigido, adquiriendo el designado por la relación entre la tensión máxima que el amplificador puede aportar y el valor de la resistencia de carga. Idealmente, el valor de la corriente que circula por la resistencia de carga se podrá calcular según se propone en la Ecuación 2-10:

$$i_{out} = V_{DD}/R_{LOAD} = \pm 45/25k = \pm 1.8mA$$

Ecuación 2-10. Cálculo ideal del valor máximo de corriente que la fuente de corriente Howland mejorada podrá proporcionar ante exigencias superiores a los requisitos funcionales que describen al sistema neuroestimulador.

La expresión mostrada en la Ecuación 2-10 será válida únicamente para valores de la resistencia de carga superiores a:

$$R_{LOAD} = V_{DD}/\, i_{out} = \pm 45/\pm 3m = 15k\Omega$$

Ecuación 2-11. Cálculo ideal del valor máximo de la resistencia de carga tal que la fuente de corriente Howland mejorada es capaz de proporcionar el máximo y mínimo valor de corriente requeridos según las especificaciones funcionales del sistema neuroestimulador.

Cabe destacar que los cálculos anteriores son meras aproximaciones que distarán en mayor o menor medida de la realidad, ya que se supone un amplificador *rail-to-rail*, es decir, un amplificador capaz de proporcionar una tensión de salida exactamente igual a la tensión de alimentación. Igualmente se ha supuesto que en la resistencia R_{3B} no cae tensión, es decir, que la tensión de salida del Howland y la tensión que cae en la resistencia de carga es la misma. No obstante, con el fin de obtener un modelo ideal de comportamiento de la fuente de corriente Howland mejorada, la siguiente Figura 2-20 detalla la corriente que podría suministrar según la impedancia de la carga conectada en su salida.

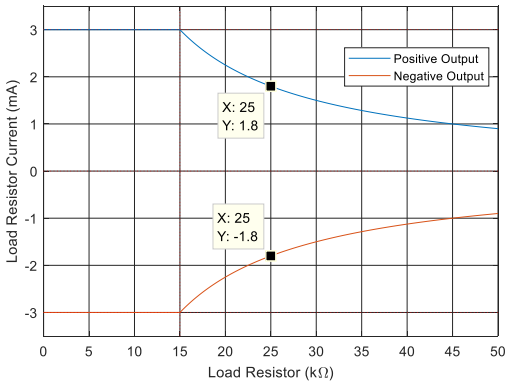


Figura 2-20. Modelado teórico de la corriente máxima y mínima que podrá proporcionar la fuente de corriente Howland mejorada ante variaciones en la impedancia de la resistencia de carga conectada en su salida.

De esta forma, el sistema ideal y de manera aproximad, cumpliría con los requisitos de diseño para cargas conectadas con impedancias menores a 15kΩ. A partir de este valor, la corriente proporcionada irá disminuyéndose racionalmente.

2.2.8 Aislamiento de las señales de trigger: Optoacoplador

Como se ha visto anteriormente, empleando convertidores DC-DC de aislamiento y amplificadores de aislamiento, se ha conseguido aislar las señales de interés y las tensiones de alimentación del sistema. Sin embargo, para conseguir aislar por completo la corriente inyectada en la carga tanto de la alimentación del sistema como del canal adyacente, es necesario aislar las señales de *trigger* encargadas de permitir o cortar la ejecución de los estímulos.

Estas señales se corresponden con pulsos digitales que pueden ser generados de forma externa o controladas de forma interna mediante el sistema Raspberry Pi. En cualquier caso, el aislamiento se ha llevado a cabo por medio de un optoacoplador.

El componente '*Vishay Semiconductors – SFH618A-2*' ha sido empleado para realizar esta operación, ya que presenta una ratio de transferencia de corriente elevado y permite ser conectado a tensiones de hasta +55V, según la hoja de datos del fabricante.

La incorporación de este elemento en el sistema puede llevarse a cabo fácilmente empleando la configuración que se muestra a continuación, en la Figura 2-21:

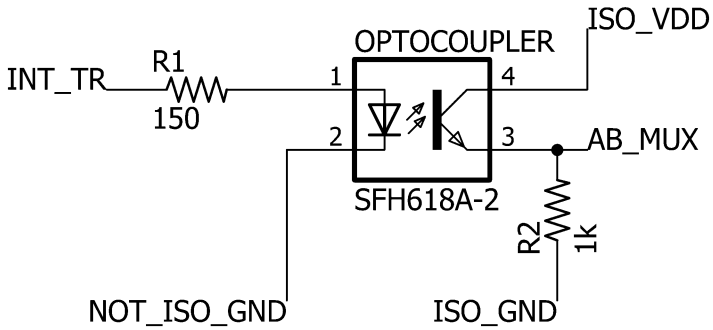


Figura 2-21. Topología de la etapa de aislamiento de las señales de trigger del subsistema hardware del sistema neuroestimulador.

El dispositivo se sitúa en la frontera que separa la zona no aislada del circuito con la zona aislada del circuito. De este modo, los pines 1 y 2 están referenciados a la tierra general del sistema; la tierra correspondiente a la tensión de alimentación principal. Por el contrario, los pines 3 y 4 quedarían referenciados a la tierra flotante propia de la alimentación de cada canal.

La tensión de +15V generada por el primero de los tres convertidores DC-DC en serie ha sido empleada para alimentar el colector del fototransistor. La resistencia R_2 desde la salida del fototransistor a la tierra aislada del canal, ha sido empleada para limitar la corriente que se dirige hacia la siguiente y última etapa, el multiplexor, el cual se explica en la siguiente sección.

Con el mismo fin anterior, la resistencia R_1 limita la corriente que recibe el diodo del optoacoplador. Superar la tensión umbral del diodo requiere que la resistencia R_1 adopte un valor pequeño como consecuencia del reducido valor de tensión digital generada por el microcontrolador, 3.3V.

2.2.9 Interruptor digital: Multiplexor

Es esta última etapa del sistema tiene lugar la sincronización de las señales que contienen los estímulos de corriente con las señales de *trigger*. Además, la arquitectura implementada permite desconectar la carga del sistema y cortocircuitar sus terminales, de forma que los terminales de salida del sistema neuroestimulador compartan el mismo potencial con objeto de eliminar las posibles tensiones y corrientes residuales que se almacenen en la carga tras la ejecución de los estímulos [9].

El dispositivo empleado que permite las operaciones anteriores indicadas es el "*Texas Instruments - CD4053BE*". Este multiplexor deberá disponer de una alimentación aislada de la alimentación global del sistema, con el fin de mantener el aislamiento que se ha llevado a cabo de las señalesde corriente y de las señales habilitadoras de ejecución de estímulos. Canal dispondrá de su propio multiplexor conectado a los ±15V aislados y generados en cada canal.

Dependiendo del valor de la señal de *trigger* (recordando que puede ser controlada de forma externa o de forma interna mediante el sistema Raspberry Pi), el multiplexor permitirá ejecutar o no los estímulos, cortocircuitando los terminales de salida en el caso de corte de ejecución.

Tanto la función lógica como las conexiones del dispositivo, se presentan a continuación en la Figura 2-22. Nótese que, para esclarecer los resultados, sólo se han representado los pines de interés.

AB	x		y	
	x_o	x_1	y_o	y_1
low	1	-	1	-
high	-	1	-	1

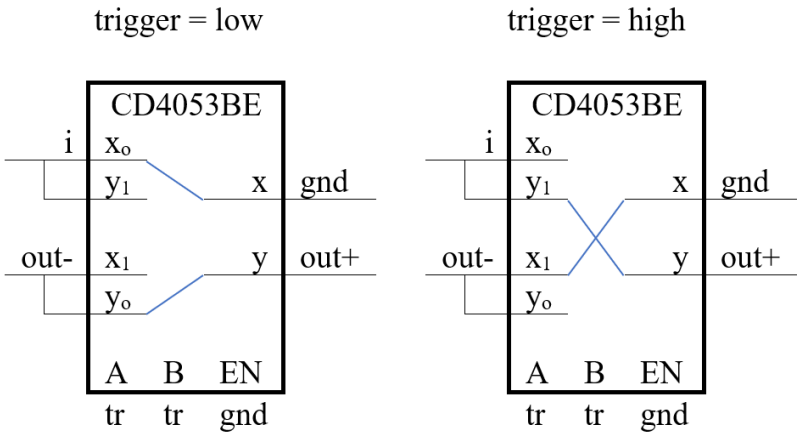


Figura 2-22. Función lógica y conexiones implementadas en el multiplexor CD4053Be del subsistema hardware del sistema neuroestimulador.

Observando la configuración anterior, el pin *enable* '*EN*' del multiplexor es conectado a la tierra flotante de cada canal, ya que el dispositivo *CD4053BE* es activo por nivel bajo. Las etiquetas '*out+*' y '*out-*' representan los dos terminales de la carga conectada a la salida del sistema neuroestimulador, la cual recibirá los estímulos, representados con la etiqueta '*i*'. Los pines '*A*' y '*B*', cortocircuitados, recibirán la señal de *trigger* aislada procedente del optoacoplador.

Para un valor de nivel alto de la señal de *trigger* (interpretado como el dar paso a la ejecución de un estímulo) la corriente será inyectada en la carga y fluirá hasta la tierra flotante propia de cada canal. Por el contrario, cuando la señal de *trigger* disponga de un valor a nivel bajo (interpretado como una parada de la ejecución del estímulo), los valores de corriente serán dirigidos directamente a la tierra flotante del canal en cuestión y los dos terminales de la carga serán cortocircuitados, con el fin de garantizar el fenómeno previamente indicado.

2.3. Vista de Desarrollo

De los tres posibles modos de funcionamiento del sistema neuroestimulador, los cuales fueron explicados en el apartado "2.1. Vista Lógica", únicamente los dos primeros precisan de un desarrollo software con el fin de poder implementar la funcionalidad que los describen. La gestión y control de la operación de ejecución de

estímulos para los modos definidos como '*Trigger In/Out*' es llevada a cabo mediante un microcontrolador, a diferencia del tercer modo de ejecución '*Waves + Trigger*', quien es la propia persona usuaria quien toma el control de la ejecución de los estímulos manualmente y en tiempo real.

El desarrollo software del sistema neuroestimulador se ha organizado en dos bloques principales: uno para la generación de la información y otro para la ejecución de la información (léase el término información como los datos que contienen las formas de onda de corriente que componen los estímulos deseados a ejecutar).

El primer bloque de generación de la información ha sido desarrollado en el software de computación numérica Matlab, y el segundo bloque de ejecución de la información ha sido implementado en la placa de desarrollo Raspberry Pi, válido para cualquier modelo de la misma.

2.3.1 Fichero de datos

Previo a la descripción de los dos bloques que conforman el subsistema software, es necesario contextualizar que la forma de comunicación entre la persona usuaria y el sistema Raspberry Pi es a través de ficheros. Mediante éstos, la persona usuaria definirá toda la información requerida por el sistema Raspberry Pi.

Para la correcta lectura e interpretación tanto del tipo de ejecución como de los parámetros que definen las formas de onda correspondientes a los estímulos, es imprescindible que la persona usuaria estructure, manualmente o mediante las funciones de Matlab que se han facilitado, la información de los ficheros tal y como se presenta a continuación.

```
# +------------------------------------------------------------+
# | NEUROESTIMULATOR                                           |
# | brief: File with stimuli definitions.                      |
# | @ author: Pablo Jimenez Fernandez < jimfermail@gmail.com > |
# +------------------------------------------------------------+

# ============================================================
# TRIGGER
# ============================================================
# 1) Setting:
# --> "yes" (with inter stimuli time)
# --> "not" (external trigger)
# 2) Examples:
# --> trigger_out = yes
# --> trigger_out = not
trigger_out = yes

# ============================================================
# CYCLES
# ============================================================
# 1) Setting:
# --> "inf" (infinite execution of stimuli)
# --> [0, ...] (finite execution of stimuli)
# 2) Examples:
# --> cycles = Inf (or inf)
# --> cycles = 0
# --> cycles = 300
cycles = 18

# ============================================================
# STIMULI STACK
# ============================================================
# 1) Format:
# --> {(Tis,) Ts, i1, i2, i3, ...}
# Tis: Inter stimuli time in usec. Only if trigger_out = yes.
#      Otherwise, this parameter should not be included.
# Fs: Sampling frequency of stimulus in Hz.
# ix: Values of current in uA.
{8111,1100,111,222,333,2095,555,666,0054,264}
{8222,2200,200,201,202,203}
{8333,3300,300,311,322,-333,344,355}
```

Observando la estructura que presenta el fichero, a primera vista puede verse cómo las líneas que cuentan con el carácter '#' se corresponden con comentarios. Dichos comentarios serán ignorados por el sistema Raspberry Pi, por lo que la persona usuaria podrá realizar todas las anotaciones que necesite sin interferir en el funcionamiento del sistema.

El fichero cuenta con un encabezado informativo y con 3 secciones diferenciadas entre sí.

La primera sección, titulada como "*TRIGGER*", representa la configuración del modo de ejecución que se pretende implementar. La sentencia "*trigger_out = yes*" indica al sistema Raspberry Pi que la ejecución de estímulos será bajo el primer modo de funcionamiento, es decir, el tiempo entre los estímulos vendrá definido por la persona usuaria. En cambio, escribiendo la sentencia como "*trigger_out = not*", se le indicará al sistema Raspberry Pi que la señal habilitadora de estímulos se recibirá de forma externa.

La sección que le continúa se titula "*CYCLES*" y se usa para designar el número de veces que se desea ejecutar una pila completa de estímulos programados. Como queda reflejado en los comentarios del fichero, la persona usuaria podrá indicar que desea una ejecución de estímulos indefinida o, al menos, hasta que la cancele manualmente, escribiendo "*cycles = inf*". La variable *cycles* deberá ser igualada al número de repeciciones que desee la persona usuaria. El número de repeticiones que la persona usuaria desee será al que deba igualar la variable *cycles*. Nótese que incluso un número de repeticiones igual a cero está permitido por el sistema Raspberry Pi, en cuyo caso, no se ejecutará ningún estímulo.

La tercera y última sección del fichero, titulada como "*STIMULI STACK*", recoge las definiciones de las formas de onda. El formato elegido para definir un estímulo es el equivalente al que se utiliza en el lenguaje de programación C para inicializar vectores. A través de los caracteres '*{*' y '*}*', la persona usuaria podrá definir el tiempo entre estímulos (solo si el sistema se encuentra funcionando bajo el primer modo de ejecución), la frecuencia de muestreo del estímulo y finalmente todas las muestras de corriente que definen la forma de onda. Utilizando el fichero de ejemplo que se expuso anteriormente, la ejecución de estímulos del sistema sería:

"El sistema neuroestimulador funcionará según el primer modo de ejecución, es decir, en el presente fichero se incluirá el tiempo que existirá entre la ejecución de dos estímulos consecutivos. En total, se repetirán 18 veces la ejecución de una pila entera de estímulos compuesta por 3 estímulos. El primero de ellos contará con un inter stimuli time igual a 8111 µs, una frecuencia de muestreo igual a 1.1kHz y las muestras de corriente serán igual a 111, 222, 333, 2095, 555, 666, 0054, 264 µA. El segundo estímulo contará con un inter stimuli igual a 8222 µs, una frecuencia de muestreo igual a 2.2kHz y las muestras de corriente serán igual a 200, 201, 202 y 203 µA. El tercer y último estímulo de esta pila, contará con un inter stimuli igual a 8333 µs, una frecuencia de muestreo igual a 3.3kHz y las muestras de corriente serán igual a 300, 311, 322, -333, 344 y 355 µA."

2.3.2 Generación de estímulos: Matlab

El objetivo final del bloque de generación de la información consiste en crear el tipo de fichero que se ha descrito en la sección anterior. Debido a la potencia de cálculo que ofrece Matlab, con objeto de simplificar y automatizar este proceso, se han desarrollado un conjunto de funciones que permiten a la persona usuaria definir las formas de onda de corriente con las que desee estimular, realizar una comprobación acerca de si los estímulos definidos se ajustan a las restricciones indicadas en las especificaciones funcionales del sistema (véase la Tabla 2-2) y, en caso afirmativo, crear los ficheros con la información definida.

El conjunto de archivos que se ha desarrollado se encuentra distribuido en determinados directorios según se muestra en la siguiente Figura 2-23:

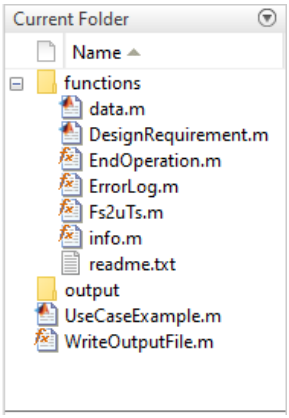


Figura 2-23. Directorio principal de trabajo de Matlab

En la Figura 2-23 se muestra el aspecto que presentaría el directorio principal de trabajo del software Matlab. En este entorno se pueden distinguir dos subcarpetas y dos archivos con extensión "*.m*" de los que se hablarán a continuación.

El primer archivo, denominado "*UseCaseExample.m*", implementa un posible caso de uso, el cual se utilizará como ejemplo de guía en las siguientes descripciones. El ejemplo que se ha decidido implementar es el correspondiente al fichero de estímulos que se mostró en la sección anterior: "2.3.1. Fichero de datos". Es decir, el resultado final tras ejecutar este *script* será obtener el fichero anterior en la denominada carpeta: "*output*".

El siguiente archivo: "*WriteOutputFile.m*", contiene la función principal que gestiona la comprobación de requisitos del estímulo y la escritura del mismo en el fichero de salida. Esta función será la que deba llamar el usuario cuando desee guardar la forma de onda que ha definido.

En el último directorio nombrado con el nombre "*functions*" se almacena el conjunto de subfunciones que el programa principal ("*WriteOutputFil.m*") necesita durante su ejecución.

Seguidamente, se mostrará el *script* de ejemplo para proceder a describir el caso de uso escogido asi como la lógica implementada en la función principal.

```
%% UseCaseExample
% \brief
% This is an example of use case.
%
% \return
% Output files with the stimuli information.

%% General parameters
% Output file name
filename = "database1";
% Number of repeats
% (set to 'inf' for infinite repeats)
cycles = 18;

%% Stimulus 1
% Current samples in uA
i = [111,222,333,2095,555,666,0054,264];
% Sampling frequency in Hz
Fs = 1100;
% Inter stimuli time in usec
% (set to '0' for trigger in mode)
Tis = 8111;
% Write output file with input information
WriteOutputFile(i, Fs, Tis, cycles, filename);

%% Stimulus 2
% Current samples in uA
i = [200,201,202,203];
% Sampling frequency in Hz
Fs = 2200;
% Inter stimuli time in usec
% (set to '0' for trigger in mode)
Tis = 8222;
% Write output file with input information
WriteOutputFile(i, Fs, Tis, cycles, filename);

%% Stimulus 3

% Current samples in uA
i = [300,311,322,-333,344,355];
% Sampling frequency in Hz
Fs = 3300;
% Inter stimuli time in usec
% (set to '0' for trigger in mode)
Tis = 8333;
% Write output file with input information
WriteOutputFile(i, Fs, Tis, cycles, filename);
```

En primer lugar, será necesario definir el nombre del fichero de salida que contendrá la información deseada. Para este ejemplo, se ha decidido usar el nombre: "*database1*". Así mismo, también se ha de definir el número de veces que se desea ejecutar la pila completa de estímulos que se está programando. Para este ejemplo, se ha decidido 18 repeticiones. Los posibles valores admitidos abarcan desde el 0, en cuyo caso no se ejecutará ningún estímulo, hasta $2^{32}/2$, número determinado por el tipo de dato usado en el software del sistema Raspberry Pi: *int32_t*. Existe, además, la posibilidad de ajustar un número de repeticiones infinita, de forma que el sistema se encuentre en ejecución indefinida hasta que la persona usuaria aborte la operación manualmente. Para dicho caso, se deberá igualar la variable usada con este fin a "*inf*" o "*Inf*".

En los siguientes pasos, tiene lugar la definición de las formas de onda y su posterior escritura en el fichero de

salida objetivo.

Los parámetros mínimos requeridos por la función de escritura quedan recogidos en la Tabla 2-4 mostrada a continuación:

Variable	Descripción	Unidad
i	Vector con las muestras de corriente del estímulo a guardar en el fichero de salida.	μA
Fs	Variable con la frecuencia de muestreo del estímulo a guardar en el fichero de salida.	Hz
tis	Variable con el tiempo entre estímulos del estímulo a guardar en el fichero de salida.	μs

Tabla 2-4. Información sobre las variables de entrada de la función "WriteOutputFile.m" implementada en la lógica de control de Matlab.

En lo que respecta a la definición del parámetro correspondiente al tiempo entre estímulos, *inter stimuli time*, si es definido igual a 0, el sistema desechará esta variable e interpretará que la ejecución será según el segundo modo de funcionamiento: *trigger_in*, es decir, mediante una señal habilitadora externa. Por otro lado, definir esta variable a cualquier valor positivo comprendido dentro del rango definido según las especificaciones, informará al sistema que la ejecución de los estímulos se realizará en base al primer modo de funcionamiento: *trigger_out*.

Tras la definición de estos 3 parámetros, se procederá a ejecutar la función principal de escritura de la información. Nótese, como dicha función recibe por parámetros de entrada los 5 definidos anteriormente: *WriteOutputFile(i, Fs, Tis, cycles, filename)*.

Desde una perspectiva global, la lógica que se ha desarrollado para la generación de los ficheros requeridos por sistema Raspberry Pi queda resumida en el digrama de flujo de la

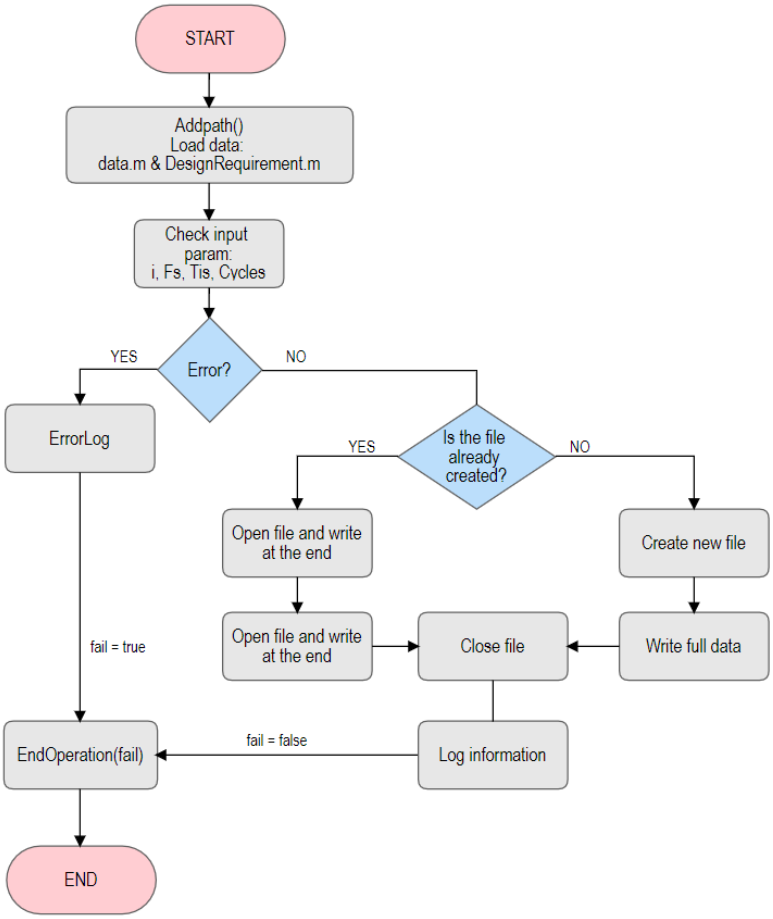


Figura 2-24.

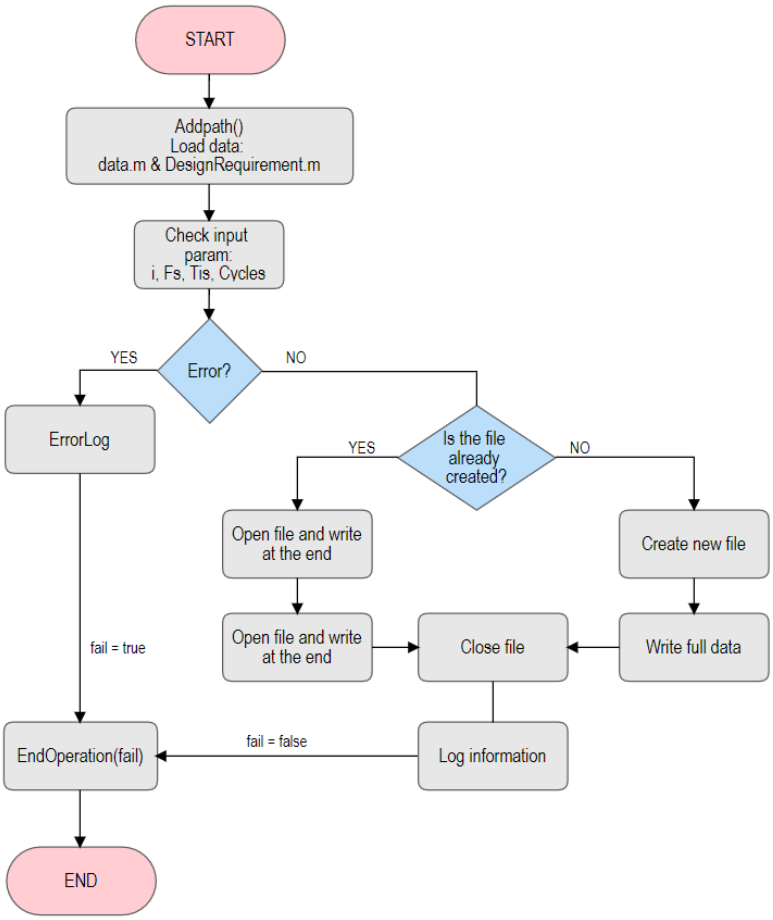


Figura 2-24. Diagrama de flujo de la función "WriteOutputFile.m" implementada en la lógica de control de Matlab.

El primer paso que implementa el diagrama de flujo anterior consiste en incluir en el directorio actual de trabajo el directorio que contiene las subfunciones necesarias: "*functions*". Una vez accesible, la función ejecutará los *scripts* "*data.m*" y "*DesignRequirements.m*". Estos 2 *scripts* permitirán poner en el *workspace* de Matlab las variables requeridas durante la ejecución del programa, variables que hacen referencia a los límites de las especificaciones funcionales y otros datos locales necesarios en la escritura de los comentarios del fichero objetivo.

Seguido a la declaración de variables, se realiza la comprobación de los parámetros que definen al estímulo a guardar. Se analiza que las muestras de corriente se ajusten al rango permitido indicado en las especificaciones del sistema neurostimulador, además de comprobar que la frecuencia de muestreo de la señal no supere el máximo soportado por el sistema. También, se comprueba que el tiempo entre estímulos se ajuste al rango permitido solo si es mayor a cero, acorde a lo citado anteriormente: "…definir esta variable a cualquier valor positivo comprendido dentro del rango definido según las especificaciones, informará al sistema que la ejecución de los estímulos se realizará en base al primer modo de funcionamiento: *trigger_out*.". Finalmente, se verifica que el número de repeticiones se ajusta al rango permitido, es decir, de 0 a $2^{32}/2$.

En caso de que la comprobación resulte un error, se le mostrarán los correspondientes mensajes por pantalla a la persona usuaria, indicándole el origen del error, para así, concluir con la ejecución del programa de escritura de forma errónea, con otro mensaje por pantalla como el que se muestra a continuación:

```
==========================
| ABORTED OPERATION...    |
| For more information execute: 'info' |
==========================
```

En caso contrario, el siguiente paso que realiza el sistema es el de comprobar si ya existe en la carpeta "*output*" un fichero con el mismo nombre que el que indicado por la persona usuaria a través del parámetro de entrada

de la función de escritura. Esta verificación permitirá, en caso de existir el fichero, no borrar los datos que existían anteriormente y tan solo escribir el vector con la información del estímulo actual, respetando el modo de ejecución y el número de repeticiones que se definió. Sin embargo, si el fichero no existe, el sistema lo creará desde cero e incluirá toda la información correspondiente al modo de ejecución, el número de repeticiones, los comentarios informativos y, por último, el vector con los datos del estímulo a guardar.

Tras la escritura de datos, tanto si existía o no el fichero, tiene lugar la llamada a la función de cerrar el fichero de salida abierto y la impresión por pantalla de un mensaje informativo en el que se indica, atendiendo al ejemplo que se está usando, lo siguiente:

File 'database1' written successfully.

```
========================
| SUCCESSFULLY OPERATION!     |
| For more information execute: 'info' |
========================
```

Este último mensaje por pantalla pone fin a la ejecución del programa encargado de la generación de la información. Como resultado final del proceso lógico, la persona usuaria dispondrá de los ficheros requeridos por la Raspberry Pi.

Cabe destacar que la función '*info.m'* se ha creado con el fin de que la persona usuaria lo ejecute en la ventana de comandos de Matlab en caso de que presente alguna duda sobre el proceso de generación de la información. Dicha función, tras ejecutarse, despliega sobre la ventana de comandos de Matlab toda la información correspondiente a los pasos a seguir para generar los archivos requeridos, además de información sobre los requisitos funcionales del sistema y descripción de las funciones y archivos que se han implementado en el software de computación numérica.

2.3.3 Ejecución de estímulos: Raspberry Pi

A continuación, entra en juego el segundo bloque que compone el subsistema software, el correspondiente a la ejecución de la información.

Esta tarea será llevada a cabo por un microcomputador, que gestionará e implementará el control de la ejecución de los estímulos. La tarjeta de desarrollo Raspberry Pi 3 Model B con el sistema operativo *Raspbian* (aunque el sistema neuroestimulador también se ha probado con la Raspberry Pi 2 Model B y es igualmente funcional) ha sido el dispositivo elegido para desempeñar dicho control, ya que, entre otros, cuenta con unas especificaciones que se ajustan a los requisitos del neuroestimulador, además de ser ampliamente conocido en el sector de la ingeniería electrónica y de control.

En primer lugar, se deberá realizar una previa y sencilla configuración del sistema operativo *Raspbian*. En esta configuración la persona usuaria deberá habilitar la comunicación I^2C provista en el sistema Raspberry Pi. Para ello, bastará con marcar como activo la casilla correspondiente a I^2C localizada en:

Inicio→Preferencias→Configuración de Rsapberry Pi→Interfaces

El siguiente paso consistirá en reservar dos núcleos de los cuatro provistos en la CPU, con el fin de ejecutar los programas de ejecución de estímulos en estos núcleos, de forma que sean las únicas tareas que se ejecuten en estos *cores*. Esta configuración permite conseguir la mínima interrupción posible por parte del sistema operativo hacia estos dos programas.

Para reservar los *cores* durante el arranque del sistema operativo, se deberá modificar el fichero "*cmdline.txt*" localizado en la carpeta *boot* del sistema, añadiendo al final de la línea: "*isolcpus=2,3*". Este comando se interpreta como la reserva de los *cores* 3 y 4 de la CPU (internamente se nombran de 0 a 3). Una posible forma de hacerlo sería ejecutando la siguiente instrucción en la consola de comandos:

~ $ sudo nano /boot/cmdline.txt

La siguiente modificación que deberá hacer la persona usuaria se corresponde con reajustar la velocidad de la

comunicación I^2C que trae por defecto Raspberry Pi, incrementándola hasta 2.5MHz. Ejecutando la siguiente instrucción, se podrá acceder al archivo que contiene dicha información:

~ $ sudo nano /boot/config.txt

Una vez que se abra el editor de texto, se deberá localizar la línea: "*dtparam=i2c_arm=on*", ya que, justo debajo, se deberá añadir y guardar la siguiente instrucción:

dtparam=i2c_baudrate=2500000

Es recomendable realizar una previa comprobación de que la comunicación I^2C se encuentra correctamente configurada. Para ello, con la PCB alimentada y conectada al sistema Raspberry Pi, se ingresará en la consola de comandos la siguiente instrucción:

~ $ i2cdetect -y 1

El dispositivo se encontrará correctamente configurado si la respuesta por parte del sistema indica que la dirección que han adquirido los dispositivos esclavos, es decir, los dos convertidores digitales – analógico, es igual a 0x60 y 0x61.

Es imprescindible comprobar que el sistema Raspberry Pi disponga de la librería "*wiringPi*", ya que el programa principal implementa funciones propias de esta librería. Se podrá comprobar que se tiene instalada ejecutando la siguiente instrucción en la consola de comandos:

~ $ gpio -v

En caso de no tener la librería, se deberá descargar e instalar desde la página principal de *Wiring Pi*.

En lo que respecta a la compilación del programa principal encargado de la ejecución de estímulos, se ha creado un archivo *Makefile* en el que se incluyen todas las opciones e instrucciones de compilación. Para compilar bastará con ejecutar el comando *make* en la consola de comandos. A continuación, se muestra el fichero de compilación que se ha desarrollado.

```
  OBJS                =    main.o    LoadDataFromFile.o
PrintStimuliData.o RunStimuliExecution.o
  SOURCE            =     main.c     LoadDataFromFile.c
PrintStimuliData.c RunStimuliExecution.c
  HEADER       =   data_raspberry_pi.h   data_stimuli.h
data_user.h       helper.h       load_data_from_file.h
print_stimuli_data.h run_stimuli_execution.h
  OUT = runstimulator
  CC   = gcc
  FLAGS    = -g -c -Wall
  LFLAGS   = -lwiringPi


  all: $(OBJS)
      $(CC) -g $(OBJS) -o $(OUT) $(LFLAGS)

  main.o: main.c
      $(CC) $(FLAGS) main.c

  LoadDataFromFile.o: LoadDataFromFile.c
      $(CC) $(FLAGS) LoadDataFromFile.c

  PrintStimuliData.o: PrintStimuliData.c
      $(CC) $(FLAGS) PrintStimuliData.c
```

```
RunStimuliExecution.o: RunStimuliExecution.c
    $(CC) $(FLAGS) RunStimuliExecution.c


clean:
    rm -f $(OBJS) $(OUT)
```

La compilación dará como resultado la creación del archivo "*runstimulator*" en el mismo directorio de trabajo. Para iniciar el proceso de ejecución de estímulos, la persona usuaria deberá ejecutar este archivo ingresando, en la consola de comandos, las instrucciones que se presentan a continuación:

~ $ sudo taskset 0x4 ./runstimulator 1

~ $ sudo taskset 0x8 ./runstimulator 2

Estos comandos tienen como propósito ejecutar el programa en los *cores* especificados, es decir, en los *cores* 3 y 4 que se reservaron inicialmente. La persona usuaria podrá seleccionar el canal en el que desea estimular mediante el parámetro de entrada que recibe la función principal *runstimulator*: "1" para ejecutar los estímulos en el canal 1 y "2" para ejecutar los estímulos en el canal 2.

Para comprobar que cada programa se ejecuta en el *core* especificado y que, además, dicho *core* cuenta únicamente con esa tarea específica, se puede ejecutar el comando "*top*" en la consola de comandos seguido del comando "*f*". A continuación, se navegará por la lista desplegada hasta la opción "*P*". Con el cursor situado en "*P*" se pulsará el comando "*d*" y finalmente la tecla '*ESC*'.

De esta forma, se desplegará un listado con todas las tareas ejecutadas en tiempo real por el sistema y el *core* en el que se encuentra ejecutándose.

Por último, antes de comenzar con la ejecución del programa principal con los comandos que se mostraron anteriormente, la persona usuaria deberá renombrar el fichero que contiene la información de los estímulos programada. El nombre designado para el primer canal es "*databasechannel1*" y el desisgnado para el segundo canal es "*databasechannel2*". Por consiguiente, la persona usuaria podrá ejecutar diferentes ficheros siempre que use los nombres reservados para ello.

Por otro lado, anteriormente se ha citado que el fichero "*runstimulator*" implementa el programa principal encargado de ejecutar los estímulos de corriente. La máquina de estados implementada durante la ejecución queda recogida en la Figura 2-25:

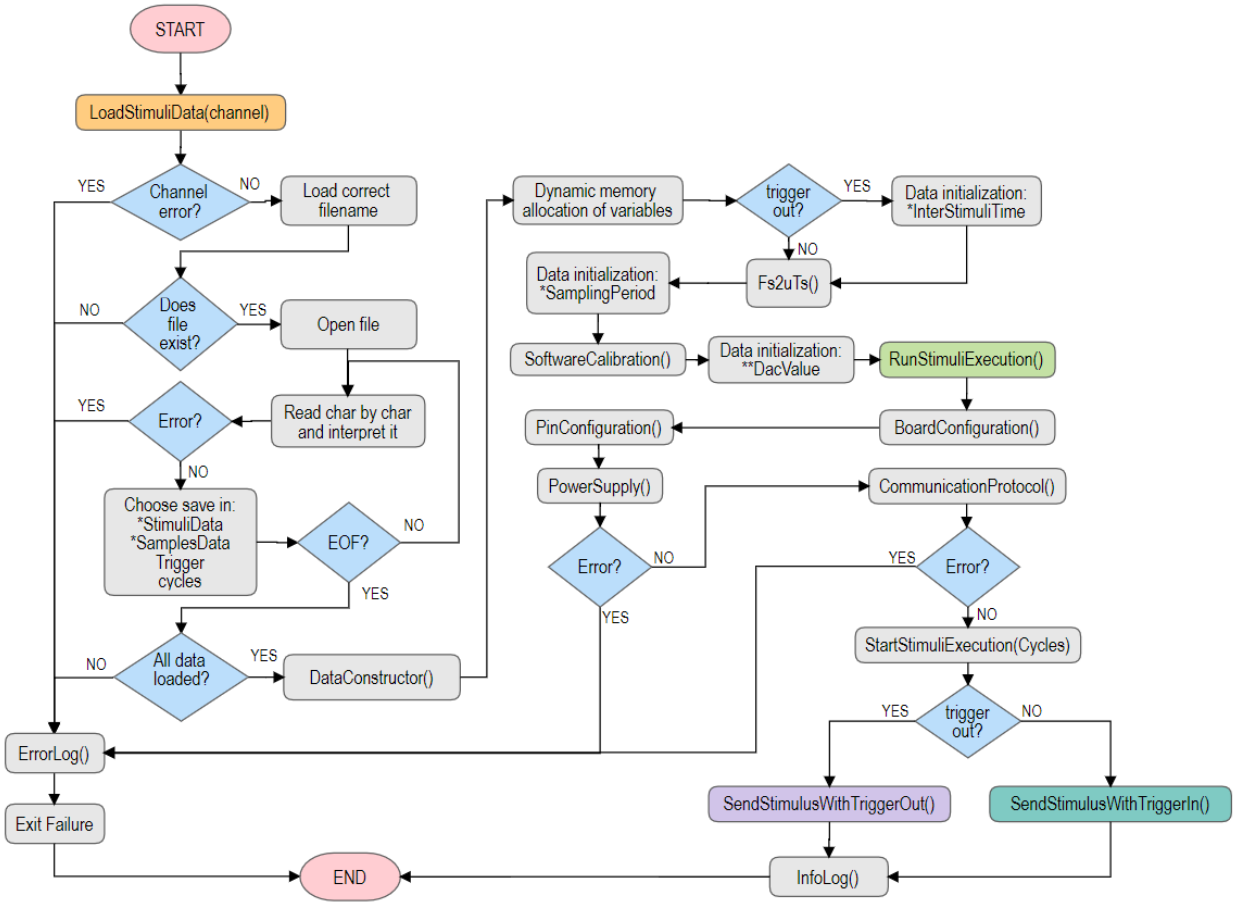


Figura 2-25. Máquina de estados implementada por el programa de ejecución de estímulos del subsistema software del sistema neuroestimulador.

La ejecución principal que se ha diseñado está constituída por dos subprocesos: uno de lectura de la información y el otro de ejecución de la información. El objetivo del primer subproceso será el de leer la información contenida en el fichero predefinido por la persona usuaria, interpretar los datos e implementar un ajuste de calibración para, posteriormente, enviar y controlar la ejecución de dichos estímulos en el segundo subproceso.

En la Figura 2-25, el primer subproceso implementado en la función "*LoadStimuliData*" recibe como parámetro de entrada el canal en el que se desea estimular. Tras comprobar que el parámetro de entrada es correcto, es decir, vale "1" o "2", se procede a cargar el nombre del fichero que se ha seleccionado. Para el caso de llamar a la función con un "1", se cargará el nombre "*databasechannel1*". Análogamente, si se introduce un "2", el sistema cargará el nombre "*databasechannel2*". Esta operación permitirá abrir correctamente el fichero con los datos que la persona usuaria ha elegido. Seguidamente, se comprueba que el fichero existe y, en caso afirmativo, lo abre. Si, por el contrario, se llama a la función con un parámetro incorrecto o si el fichero no existe, se imprimirá un mensaje por pantalla indicando del error y se pondrá fin a la ejecución principal.

Una vez que el fichero ha sido correctamente abierto, se procede a la lectura e interpretación de todos los caracteres que lo componen. Citado anteriormente, el sistema descartará todos los comentarios que encuentre, es decir, se descartarán las líneas completas que comiencen por el carácter '#'. Los datos leídos que sean correctos se irán almacenando en el vector definido como "*StimuliData*". Igualmente, se guardará el modo de ejecución "*TriggerOut*" y el número de veces que se repite la ejecución "*cycles*". Nuevamente, la ejecución principal se detendrá e informará por pantalla si la información contenida en el fichero es errónea.

En todo momento de la captación de información y si se precisa, el sistema irá ampliando dinámicamente el tamaño de los vectores "*StimuliData*" y "*SamplesData*". En este último vector, se irán almacenando el número total de muestras de corriente que componen a cada estímulo.

Cuando se llegue al final del fichero, designado como "*EOF: End of File*", se comprueba que todos los

parámetros necesarios para la ejecución (el modo de ejecución, el número de repeticiones y los datos que definen a los estímulos) han sido correctamente leídos y copiados en las variables oportunas. En caso de faltar algún parámetro, el sistema notificará por pantalla y se pondrá fin a la ejecución del programa principal.

Con todos los datos cargados, se procede a llamar a la función "*DataConstructor*". Esta función tiene como objetivo desglosar y organizar la información almacenada en caso de que la persona usuaria desee ampliar la funcionalidad de la ejecución o crear nuevas funciones, para que disponga fácilmente de las variables que definen a los estímulos.

En la función del constructor de datos, en primer lugar, se gestiona la redimensión dinámica de los vectores que contienen los datos de los estímulos, a partir del conjunto de datos almacenados en el vector *StimuliData*. Estos nuevos vectores forman parte de una estructura contexto global, diseñada para almacenar la información que define a los estímulos. El constructor se encargará de inicializar este contexto.

Inicialmente, el constructor comprueba la información correspondiente al tipo de ejecución que se ha leído del fichero. Solo en caso de que la ejecución esté programada para funcionar bajo el primer modo de ejecución o *trigger_out*, el constructor interpretará que, de los conjuntos de estímulos almacenados en *StimuliData*, el primer campo se corresponderá con el tiempo entre estímulos definido por el usuario. Por consiguiente, el constructor almacenará en el mismo vector todos los tiempos entre estímulos. Seguidamente, empleando un conversor de frecuencia de muestreo a periodo de muestreo en microsegundos, la función "*Fs2uTs*", se irán almancenando los períodos de muestreo de los estímulos otra variable del contexto. Cabe destacar que la frecuencia de muestro de cada estímulo se correspondía con el segundo valor del conjunto de estímulos almacenados en el vector de datos *StimuliData*. De la misma manera, el constructor guardará las muestras de corriente convertidas a los valores correspondientes del convertidor digital – analógico haciendo uso de una función de calibración.

La función de calibración busca encontrar el valor del convertidor digital – analógico (comprendido entre 0 y 4095) que haga que el neuroestimulador proporcione a la salida el valor de corriente más cercano al predefinido por la persona usuaria.

Para el algoritmo de cálculo que implementa dicha función ha sido necesaria la elaboración de un modelo teórico que se aproxime al comportamiento del sistema neuroestimulador, por lo que ha sido imprescindible caracterizarlo.

En el experimento de caracterización del sistema se ha excitado la entrada del sistema barriendo el rango completo de entrada del convertidor digital – analógico de 0 a 4095 y se ha medido la corriente de salida para una carga igual a 1kΩ.

Se ha experimentado con diferentes métodos de aproximación con el objetivo de encontrar el modelo que más se ajuste al comportamiento real del sistema. En la Figura 2-26 se muestran los errores existentes entre el modelo real y el teórico para los diferentes métodos de aproximación.

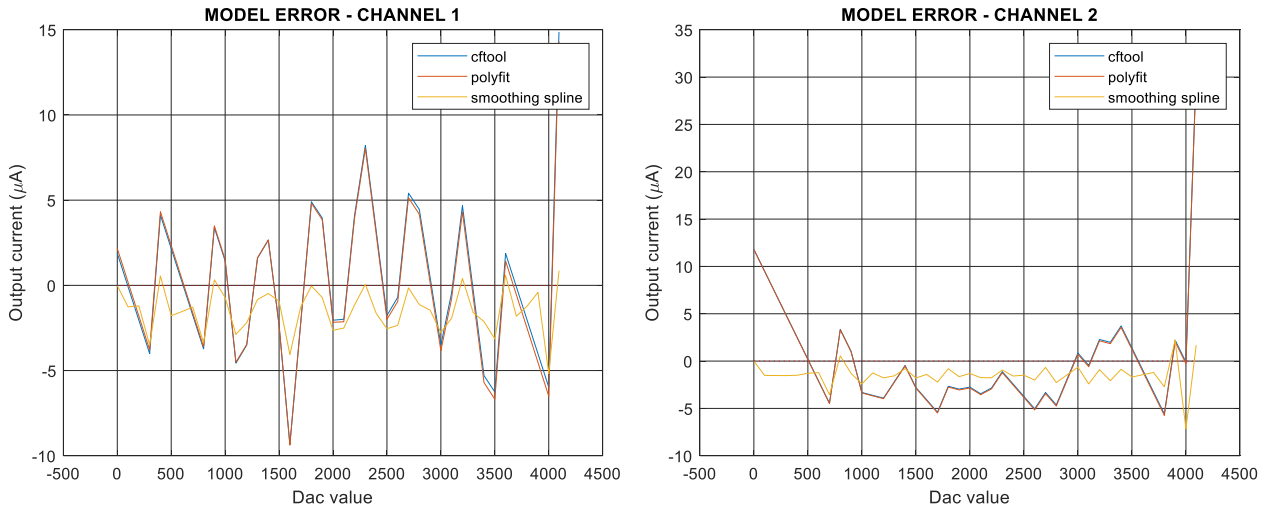


Figura 2-26. Error de los diferentes métodos de aproximación empleados en la elaboración del modelo del sistema neuroestimulador para ambos canales.

Como puede comprobarse en los resultados anteriores, el modelo de menor error es el obtenido al hacer la aproximación matemática definida como *Smoothing Spline* [14].

Llegados a este punto, tiene lugar el fin del primer subproceso del programa principal de ejecución de estímulos y se abre paso al segundo, al envío de los datos que se han preparado anteriormente.

En la Figura 2-25, el segundo subproceso se reprenta con la función "*RunStimuliExecution*". En esta función, tienen lugar dos tareas principales: realizar la configuración de los puertos de la placa del sistema Raspberry Pi y enviar los datos a través de ella, hacia la PCB.

La configuración de la placa abarca la configuración de los pines del sistema Raspberry Pi, la comprobación de la fuente de alimentación y la configuración del protocolo de comunicación, que para nuestra causa es el I^2C.

Configurar un pin como pin de tipo *input* o de tipo *output* dependerá en su totalidad del tipo de ejecución que se esté implementado. El pin será de tipo *output* si la ejecución es según el primer modo, dado que será el microcomputador quien envíe al exterior las señales de *trigger* gestionadas internamente. Por el contrario, el pin será de tipo *input* cuando el sistema funcione según el segundo modo de ejecución, caso en el que las señales habilitadoras se reciben externamente.

Para la comprobación de la fuente de alimentación de los canales, simplemente se verificará el *flag* de aviso explicado en el punto "2.2.1. Interruptor START". El hecho de que el pin que gestiona el *flag* se encuentre a nivel bajo, hará que el sistema interprete que el canal no disponde de fuente de alimentación, por lo que pondrá fin a la ejecución del programa principal mostrando un mensaje por pantalla con el aviso del error de alimentación.

Si el sistema se encuentra correctamente alimentado, se prepara la comunicación I^2C con la dirección particular de cada convertidor digital – analógico que, según se mencionó anteriormente, debe ser 0x60 para el *dac* del canal 1 y 0x61 para el *dac* del canal 2.

En caso de ausencia de error, se procede, en última estancia, a la ejecución de los estímulos. La pila de estímulos, es decir, el conjunto de estímulos programados por la persona usuaria será repetido tantas veces como se haya indicado en la variable "*cycles*".

Según el modo de ejecución, el sistema llamará a la función "*SendStimulusWithTriggerOut*" o a la función "*SendStimulusWithTriggerIn*".

Ambas funciones se encargarán de enviar por I^2C a la frecuencia de muestreo específica, los valores del convertidor que se prepararon previamente. Así mismo, durante toda la ejecución de un estímulo, es decir, durante el envío de todas sus muestras, el sistema pondrá a nivel alto los pines correspondientes a las señales de *trigger*, con el fin de que las muestras de corriente atraviesen el multiplexor. Cuando cese el envío de dichas muestras, las señales de *trigger* volverán a ponerse a cero para cortar el envío de señal.

Para el primer modo de ejecución se han implementado una serie de *timers* o funciones de contabilización del tiempo con objeto de poder ejecutar correctamente los estímulos en los tiempos entre estímulos que definió la persona usuaria inicialmente. Por el contrario, para el segundo modo de ejecución, al comienzo de cada ejecución de estímulos tiene lugar una llamada a una función que se mantiene en espera de recibir las señales de *trigger* externas al dispositivo.

Por otro lado, en el momento en el que el sistema detecte cualquier error, se parará el envío de muestras y se pondrá fin a la ejecución del programa, indicándose con un *log* por la consola de comandos el error acaecido.

Cuando la ejecución de todos los estímulos programados concluya satisfactoriamente, se mostrará un *log* por la consola de comandos con la información correspondiente a que la ejecución terminó satisfactoriamente y sin errores. En caso de contar con una ejecución infinita, el usuario podrá cancelar la ejecución de los estímulos con el comando *ctrl+c*.

En la Figura 2-27 se recogen las máquinas de estado implementadas para el envío de las muestras de los estímulos.

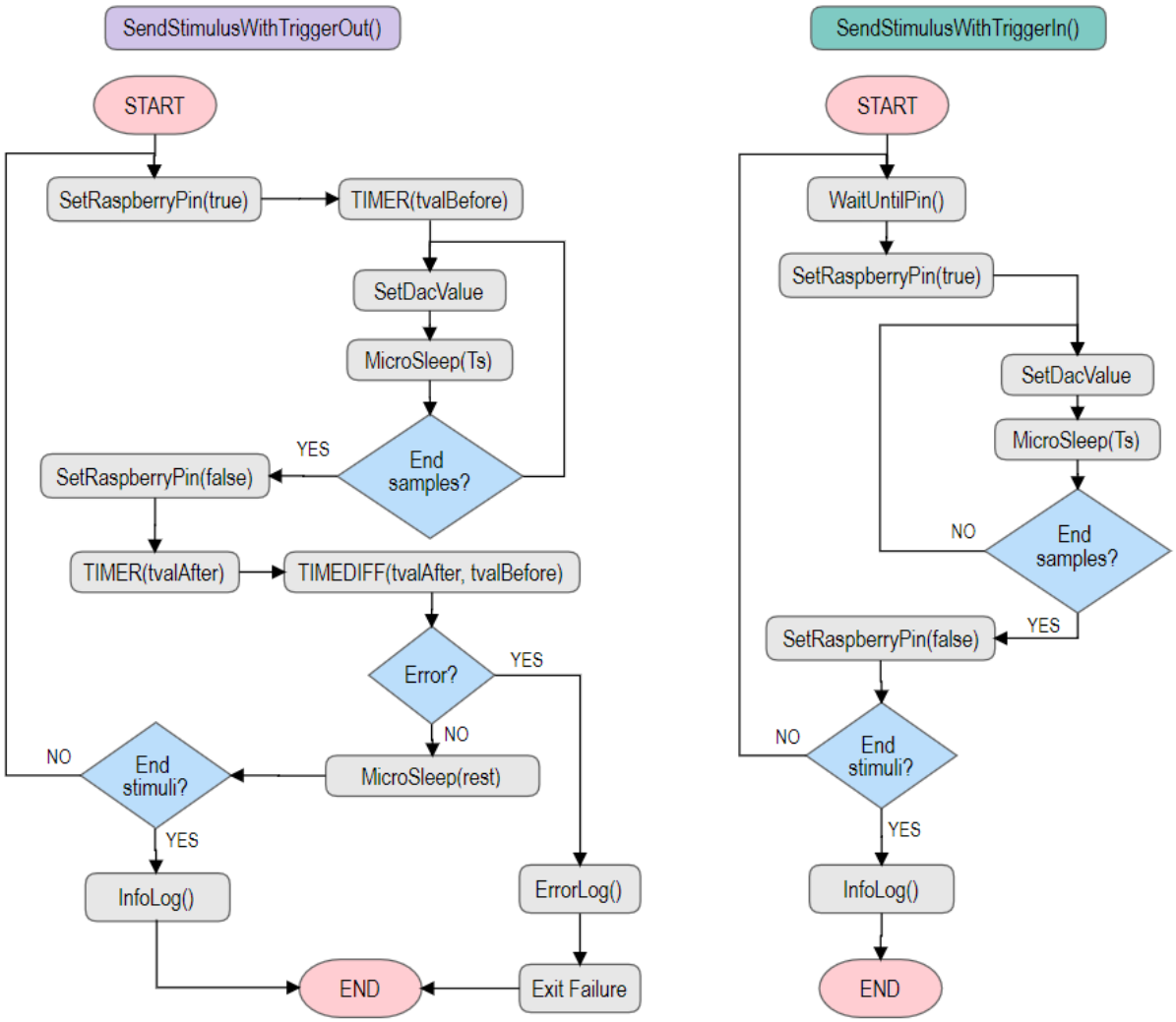


Figura 2-27. Máquina de estados implementada por las funciones de envío de estímulos del subsistema software del sistema neuroestimulador.

2.4. Vista de Procesos

Para los modos de ejecución en los que intervenga el procesamiento y control del sistema Raspberry Pi, es decir, el primer y segundo modo de ejecución, la persona usuaria manualmente o a través de funciones creadas en Matlab definirá en los ficheros nombrados como '*databasechannel1*' y '*databasechannel2*' tanto la pila de estímulos que desee ejecutar como los parámetros necesarios que definan el modo de ejecución de los estímulos. Tras la correcta verificación, mediante funciones vía Matlab del cumplimiento de los requisitos funcionales del sistema neuroestimulador y de la correcta sintaxis de los ficheros generados que contienen la información requerida, estos archivos deberán ser incluídos en el mismo directorio del programa de ejecución de estímulos de la microSD que contiene el sistema operativo y los datos del sistema Raspberry Pi. Este procedimiento de localización de archivos se podrá llevar a cabo vía Internet o manualmente por la persona usuaria, copiando y pegando los archivos en la ruta especificada. De esta forma, el programa encargado de la ejecución de estímulos podrá hacer uso de estos datos que han sido generados.

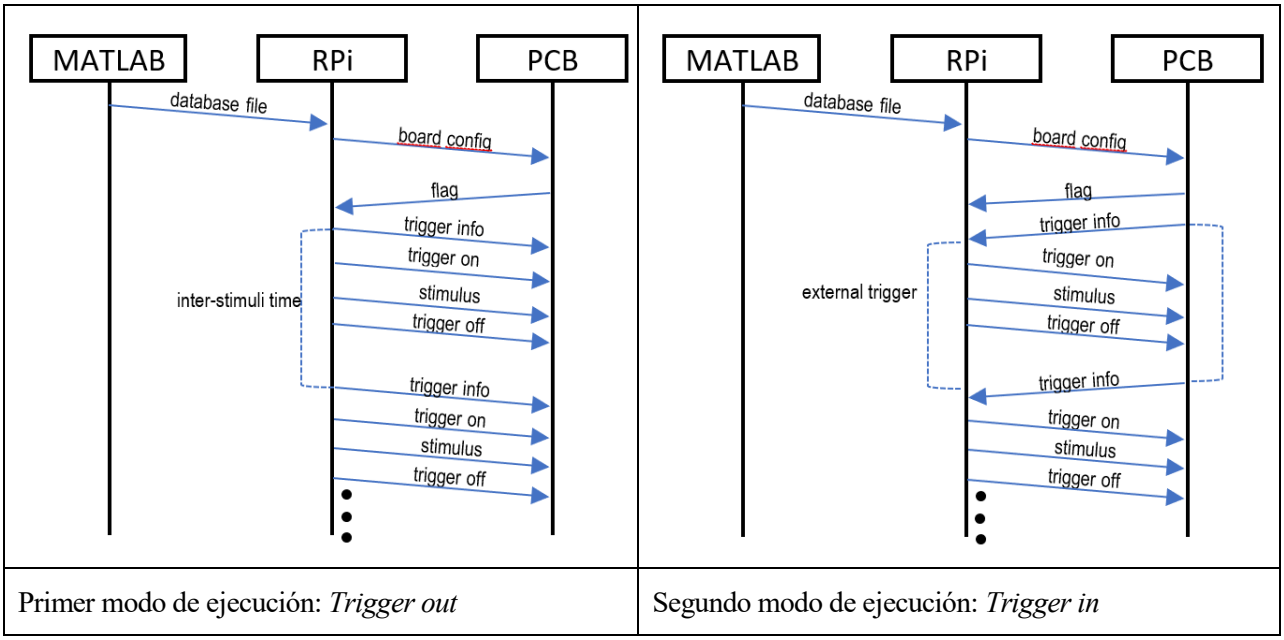
Consecutivamente, se ejecuta en el microcomputador el programa principal de ejecución de los estímulos. La lógica programada extrae e interpreta la información contenida en dichos ficheros definidos por la persona usuaria, prepara el sistema y carga los estímulos en la SRAM del sistema Raspberry Pi, quedando a la espera de recibir un *flag* de aviso por parte de la PCB, el cual indica que el sistema hardware dispone de suministro y se encuentra preparado para recibir y procesar los estímulos.

Una vez que el proceso comienza, el sistema Raspberry Pi envía de forma secuencial la pila de estímulos que la persona usuaria ha programado junto con las determinadas señales de *trigger* que permiten el envío externo

de los estímulos hacia la carga o el corte de la ejecución, cortocircuitando los terminales de la carga conectada. De este modo, según el primer o segundo modo de ejecución, el microcomputador se encarga de controlar y gestionar la información de las señales de *trigger* en los pines externos de la PCB, designados como "*ETR_A*" y "*ETR_B*".

La ejecución de estímulos continuará indefinidamente o hasta haber ejecutado las pilas de estímulos de ambos canales tantas veces como haya indicado la persona usuaria en los archivos de datos generados. La persona usuaria, en cualquier momento, podrá cortar el suministro de energía en la PCB accionando el mismo interruptor usado para comenzar con la ejecución. El corte de suministro será interpretado por el microcomputador, que dejará de enviar estímulos, poniendo fin a la ejecución del programa principal.

El esquema de los eventos que tienen lugar en el proceso de ejecución de estímulos son los mostrados a continuación, en la Figura 2-28:



Primer modo de ejecución: *Trigger out*	Segundo modo de ejecución: *Trigger in*

Figura 2-28. Flujo de eventos que tienen lugar en el sistema neuroestimulador durante la ejecución de la pila de estímulos programados y definidos inicialemente por la persona usuaria.

Por el contrario, en el tercer modo de ejecución, denominado anteriormente como "*Waves* + Trigger", se prescinde del uso del sistema Raspberry Pi y la persona usuaria inyecta las señales de tensión directamente sobre la PCB, en los *jumpers* designados para ese fin. El control de las señales de *trigger* lo deberá llevar a cabo externamente, inyectando, también, dichas señales de control de ejecución directamente sobre la PCB.

2.5. Escenario

La última vista de la arquitectura 4+1 tiene como fin mostrar los resultados obtenidos tras implemetar varios casos de uso en el sistema neuroestimulador.

2.5.1 Caracterización del sistema neuroestimulador

En primer lugar, se ha considerado realizar una caracterización del sistema neuroestimulador. Para ello, se ha barrido el rango de corriente de salida para el que se ha diseñado: ±3mA. La carga que se ha seleccionado para dichos experimentos ha sido una resistencia de 1kΩ.

Para medir la corriente que circula por la carga, se ha medido la tensión media que cae sobre ella con un osciloscopio portátil, es decir, aislado de la fuente de alimentación, con el fin de respetar el aislamiento que proporciona el sistema neuroestimulador. Las medidas se han representado usando Matlab. En la Figura 2-29 se exponen los resultados.

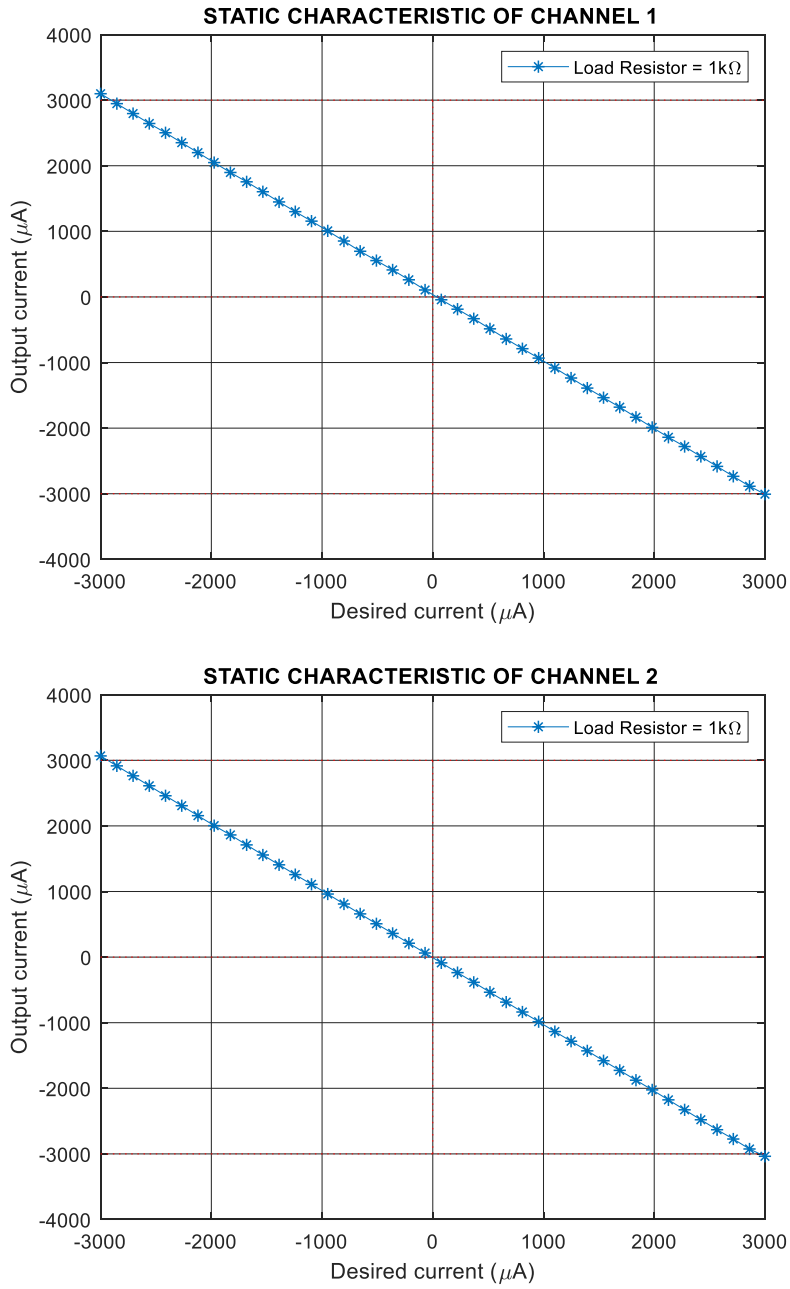


Figura 2-29. Resultados de la caracterización del sistema neuroestimulador empleando una resistencia de carga igual a 1kΩ

Para los siguientes experimentos, más realistas, se ha colocado una resistencia de 100kΩ en los pines de salida de cada canal, a modo de simular la impendancia que presentaría un electrodo común. Tras aplicar los estímulos seleccionados a conciencia sobre dicha carga, las formas de onda resultantes, en este caso señales de tensión, serán capturadas con un osciloscopio portátil con objeto de respetar nuevamente el aislamiento de la señal.

2.5.2 Estímulo tradicional: Onda bifásica

En este caso de uso, los estímulos a ejecutar consistirán en formas de onda bifásicas de diferentes niveles de amplitud, con un retraso específico entre las dos fases y con una duración de fase determinada. Dichos estímulos serán controlados por un *trigger* externo y se ejecutarán en un solo canal. El experimento contará con un total de 50 repeticiones.

La señal estará constituida por cuatro pulsos, uno de 200µA de amplitud, otro de -200µA, el siguiente de 100µA y el último de -100µA. La forma de onda resultante para el caso de los dos primeros pulsos es la que se muestra en la Figura 2-30:

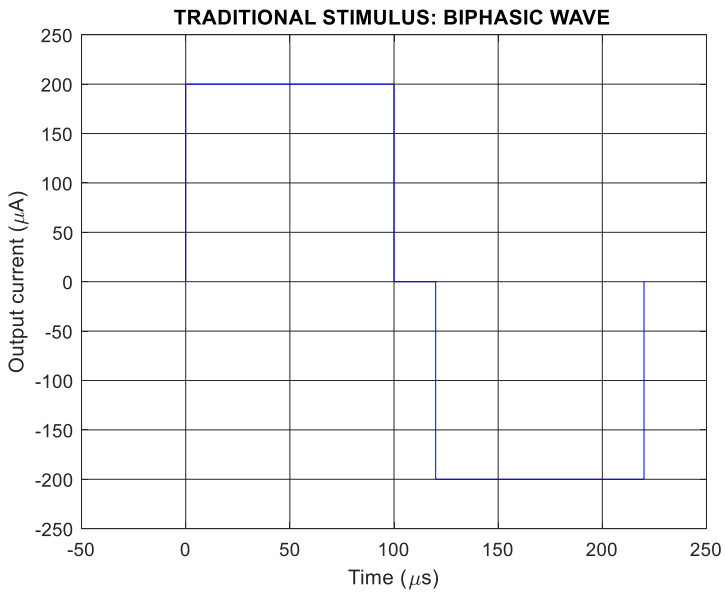


Figura 2-30. Representación de la forma de onda bifásica del estímulo tradicional empleado en el primer caso de uso.

La forma de onda capturada es la que se presenta en la Figura 2-31:

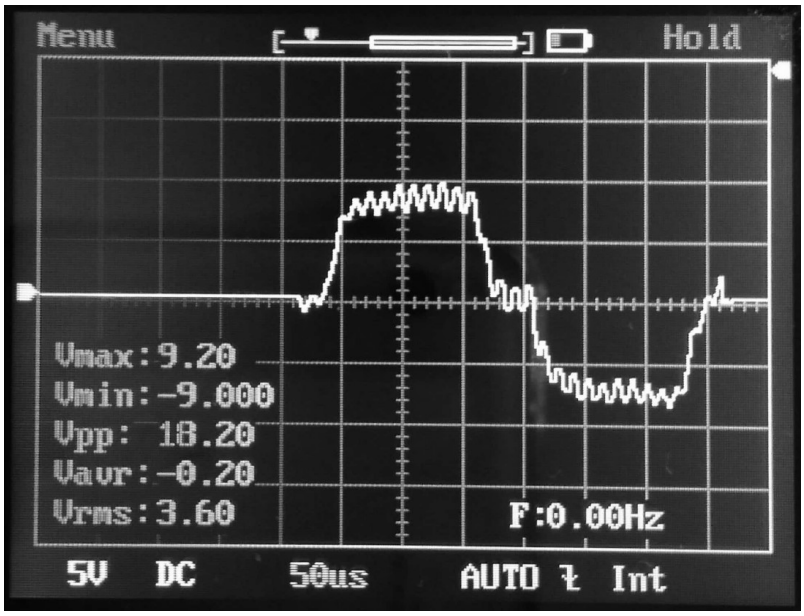


Figura 2-31. Resultado de la medición de la forma de onda bifásica del estímulo tradicional empleado en el primer caso de uso.

2.5.3 Resonancia estocástica: Ruido

La estimulación con señales ruidosas puede ayudar a la activación de neuronas, ya que en determinados puntos de esos niveles aleatorios puede aportarse la energía necesaria para superar el umbral mínimo requerido [15]. A consecuencia de lo anterior, se procederá a estimular con ruido. La duración de cada estímulo será de 300ms e implementarán el primer modo de ejecución, es decir, se definirá el tiempo que habrá entre cada estímulo. Dicho tiempo será de 1013ms y cada vez que se ejecute un estímulo, el sistema lanzará una señal de *trigger* al exterior. Los estímulos contarán con una amplitud máxima igual a 50µA y se repetirán un total de 50 veces, empleando un solo canal del neuroestimulador. La forma de onda capturada es la que se presenta en la Figura 2-32:

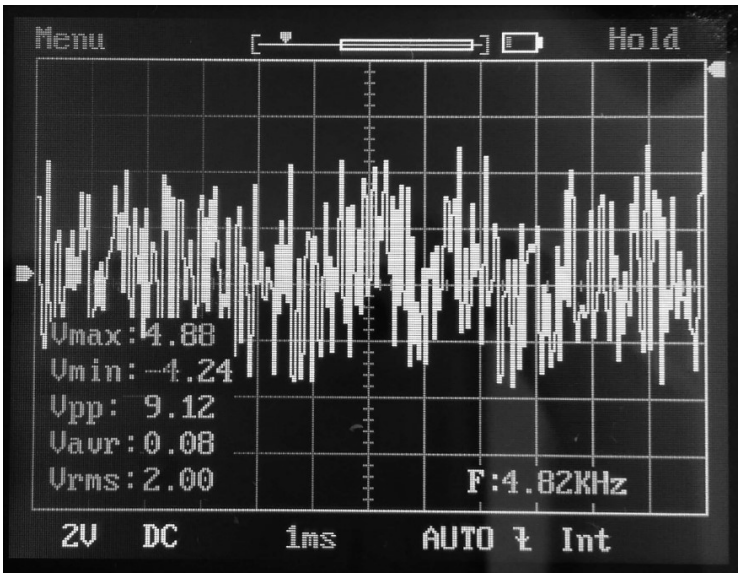


Figura 2-32. Resultados de la medición del ruido empleado como estímulo en la resonancia estocástica del segundo caso de uso.

2.5.4 Estimulación profunda no invasiva: Onda sinusoidal

El experimento consistirá en generar dos sinusoides; una en cada canal del neuroestimulador. La primera será a 7.5kHz y la segunda a 7.5kHz. Ambas contarán con una amplitud de pico igual a 200µA y una duración de 500ms con otros 500ms sin estimular. El tiempo entre estímulos se ha fijado en 1013ms. El experimento se repetirá 50 veces. La forma de onda capturada es la que se presenta en la Figura 2-33:

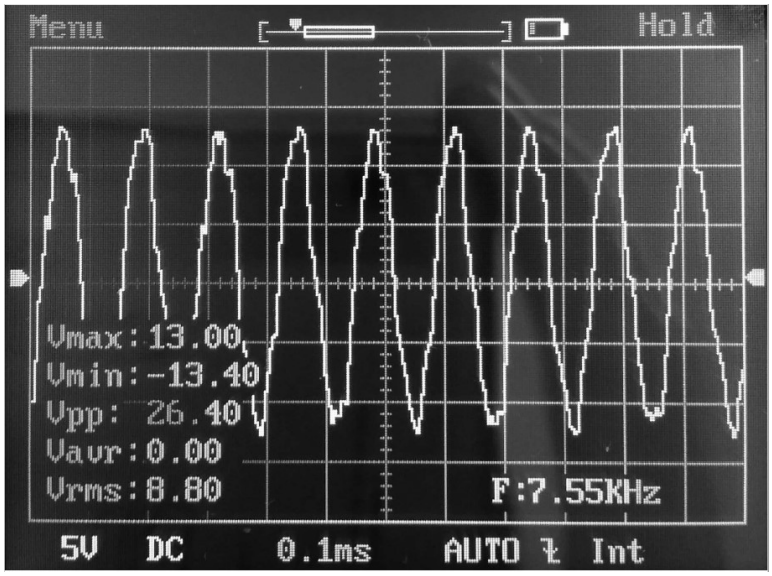


Figura 2-33. Resultados de la medición de la onda sinusoidal de 7.5kHz empleada como estímulo en la estimulación profunda no invasiva

2.5.5 Tercer modo de ejecución: Generador de tensión externo

En el último experimento se ha puesto a pruebas el tercer modo de ejecución. Desconectando los *jumpers* provistos en la PCB e inyectando señal a través de un generador de ondas de tensión, se ha decidido estimular con una onda sinusoidal de la frecuencia máxima indicada en los requisites de diseño: 30kHz.

Los resultados quedan representados en la Figura 2-34:

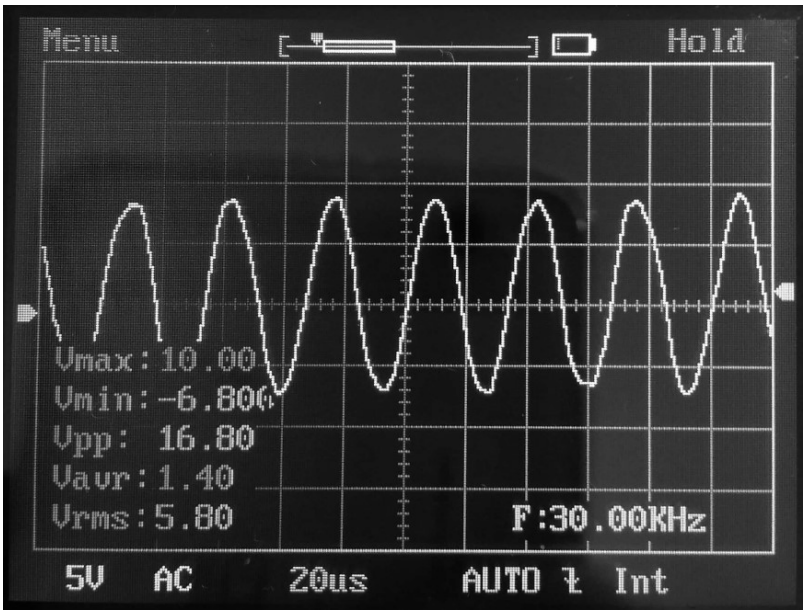


Figura 2-34. Resultados de la medición de la onda sinusoidal de 30kHz empledad en el caso de uso que implementa el tercer modo de ejecución.

3 DISCUSIONES Y CONCLUSIONES

El capítulo actual tiene por objeto describir los diferentes análisis de los procedimientos y métodos alternativos al implementado, así como las implicaciones acaecidas en el desarrollo y puesta en marcha del sistema neuroestimulador. Por último, se expondrá una breve valoración de los resultados obtenidos.

Como alternativa al sistema Raspberry Pi, se ha analizado la placa Arduino Due como posible microcontrolador del subsistema software, debido a su reducido coste y por encontrarse muy extendida globalmente.

Este dispositivo cuenta con una capacidad de memoria SRAM igual 96kB y 512kB para la memoria FLASH. A priori, puede pensarse que las capacidades de memoria disponibles resultarán insuficientes para nuestra causa. No obstante, se ha visto interesante realizar un breve estudio de la memoria con el fin de obtener datos objetivos.

Con la implementación de una sencilla función en Matlab se consigue estimar de forma aproximada la cantidad de memoria que ocuparán los estímulos. La función permite operar en tres escenarios diferentes:

- Caso 1: Cálculo para un número específico de estímulos.
- Caso 2: Cálculo para un número máximo de muestras de cada estímulo.
- Caso 3: Cáclulo para una pila específica de estímulo.

A continuación, en la Tabla 3-1 se muestran los resultados de los tres escenarios.

Caso 1	Caso 2	Caso 3
> For 5000 samples and with a program code that occupies 10%: + Maximum stimuli number: 20 + FLASH used: 460.81 Kb (90.00%) + SRAM used: 30.08 Kb (31.33%)	> For 100 stimuli and with a program code that occupies 10%: + Maximum sample number: 1049 + FLASH used: 460.97 Kb (90.03%) + SRAM used: 13.70 Kb (14.27%)	> For 100 stimuli with 65000 samples per stimulus and with a program code that occupies 10%: + FLASH used: 25441.83 Kb (4969.11%) + SRAM used: 263.51 Kb (274.49%)

Tabla 3-1. Resultados del cálculo del tamaño de los estímulos especificados para la placa Arduino Due.

Con los resultados obtenidos, se corrobora lo que se estimó inicialmente. La tarjeta Arduino Due no cuenta con la memoria suficiente requerida por las especificaciones funcionales del proyecto actual.

El siguiente análisis que se ha realizado consiste en expandir la memoria del microcontrolador empleando una tarjeta microSD externa y un lector de tarjetas conectado a la placa mediante comunicación SPI.

En el análisis, se ha medido el tiempo que emplea el microcontrolador en situar en memoria SRAM las muestras que lea tanto en el caso de usar la memoria externa como en el caso de usar la propia memoria FLASH. Véase la Figura 3-1.

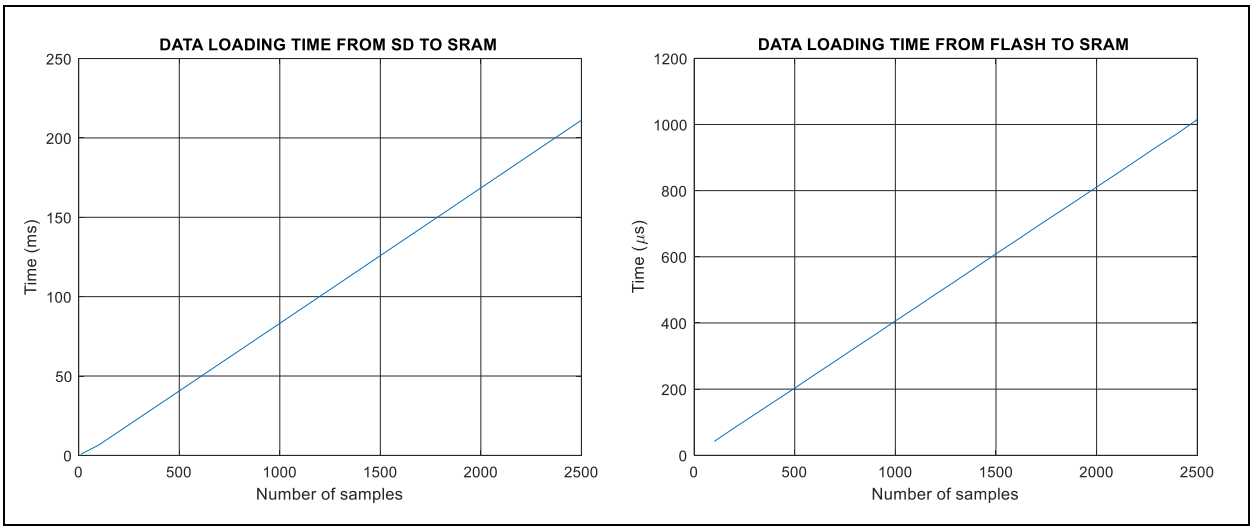


Figura 3-1. Análisis del tiempo empleado por el microcontrolador Arduino Due para cargar en SRAM un conjunto de muestras procedentes de una memoria externa microSD y de la propia memoria FLASH.

A la vista de los resultados, puede comprobarse que los tiempos de carga son excesivamente altos comparados con los tiempos de ejecución del sistema neuroestimulador.

Este breve análisis indica que la tarjeta Arduino Due no resulta ser una buena alternativa como microcontrolador para el sistema actual.

Seguidamente y trasladando el análisis temporal anterior al sistema Raspberry Pi, se ha visto necesario optar por el lenguaje C en lugar del lenguaje Python en el desarrollo software del neuroestimulador.

El hecho de que Python sea un lenguaje de programación interpretado da como resultado que las funciones de temporización del sistema no sean todo lo precisas que se requieren para el proyecto actual. La combinación Python + C, empleando C solo en el desarrollo de las funciones de temporización mediante *wrappers* y *ctypes*, tampoco se ajusta a los requisitos.

Las funciones de temporización implementadas en lenguaje de programación de bajo nivel, C, ofrecen mejores resultados, aunque no terminan de adecuarse por completo a lo solicitado.

Con el fin de evitar que otras tareas de procesamiento interrumpan la ejecución de los temporizadores, se ha probado a reservar los núcleos 3 y 4 de la CPU, citado en el apartado "2.3.3. Ejecución de estímulos: Raspberry Pi", y ejecutar los temporizadores en dichos núcleos reservados, operación que otorga mejores resultados, aunque, nuevamente sin alcanzar los objetivos.

Finalmente, se ha visto que la gestión más precisa del tiempo se consigue con la implementación de un bucle *while* ocupado, *busy loop*, llegando hasta poder medir y gestionar varias unidades de microsegundos. Como contrapartida, el gasto de procesamiento por parte del sistema implica que las funciones que hacen uso de estos *timers* se encuentren ejecutadas en los *cores* reservados.

Por otro lado, a la vista de los resultados obtenidos en los casos de uso realizados en el apartado "2.5 Escenario", es conveniente reflexionar sobre el diseño implementado en la construcción del sistema neuroestimulador.

En un principio, viendo el comportamiento del neuroestimulador en las características estáticas expuestas en el apartado "2.5.1 Caracterización del sistema neuroestimulador" puede comprobarse que el sistema, configurado para una carga de 1kΩ, puede ofrecer el rango completo de corriente para el que se ha diseñado: ±3mA.

Comparando la corriente de salida con la que se espera, puede apreciarse que el sistema cuenta con un ligero *offset*. Esta desviación no supone un gran problema, ya que, como se citó anteriormente, se ha realizado una calibración via software que permite corregir estos valores. Cabe destacar, que la caracterización se realizó sin implementar la calibración software.

El siguiente aspecto a destacar, analizando los resultados del resto de casos de uso del apartado "2.5 Escenario", es que el sistema neuroestimulador, para los dos primeros modos de ejecución, es capaz de ofrecer formas de corriente con una frecuencia máxima de 7.5kHz. Puede comprobarse entonces que no se ha

conseguido cumplir con los requisitos de diseño expuestos: "generar formas de onda de corriente arbitrarias hasta una frecuencia de 30kHz".

En este hecho, aunque solventable si se emplea el tercer modo de ejecución, uno de los factores limitantes de la velocidad es el software implementado en el sistema Raspberry Pi. Concretamente en la velocidad de escritura de cada muestra mediante la comunicación I^2C.

Se ha medido el tiempo que tarda el sistema Raspberry Pi en colocar una sola muestra en el convertidor analógico – digital y el resultado ha sido que emplea 24μs en realizar la operación, tiempo que equivale a una frecuencia de 41.667kHz. Con dicha frencuencia, y empleando el teorema de Nyquist, podremos decir que la señal más rápida que se podrá ejecutar será de 20.883kHz, suponiendo que se encuentra constituida únicamente por 2 muestras. De esta manera, cuantas más muestras compongan la señal original a muestrear, menor frecuencia se conseguirá. Como posible alternativa a esta limitación, podría reconfigurarse el sistema para sustituir el protocolo de comunicación I^2C por el protocolo SPI, que en general, presenta velocidades superiores.

Cabe decir que, en la configuración interna del sistema Raspberry Pi, se ha cambiado el valor por defecto que existía en la velocidad de la comunicación I^2C. El valor de reemplazo ha sido elegido de forma que el sistema envíe a la máxima velocidad de manera constante, sin variaciones en ese tiempo de escritura. De esta forma, el valor óptimo ha sido 2.5Mhz. Valores superiores al anterior provocaban variaciones incontroladas en el tiempo de escritura de cada muestra.

Una particularidad que necesariamente debe ser citada por la importante limitación que supone para el sistema neuroestimulador es el comportamiento que presenta el canal 2. Se ha medido experimentalmente que dicho canal denota un comportamiento que dista del que presenta su análogo, el canal 1. En el experimento realizado, el empleo de una carga igual a 100kΩ y valores de corriente entorno a ±200μF (aplicando la conversión que implementa el sistema: ±20V requeridos por el opamp de 100V) llevaban al opamp de alto voltaje a la zona de saturación, lo que provoca la saturación del canal y la imposibilidad de proporcionar lo esperado.

Otro aspecto que merece especial atención ha sido el ruido presente en las formas de onda generadas por el neuroestimulador. Tras realizar diferentes análisis, con objeto de identificar la fuente de ruido, se ha visto que éste se incrementa a medida que se disminuye el valor de la carga conectada a la salida del sistema y que, además, adquiere mayor fuerza en la etapa correspondiente a la fuente de corriente Howland. No obstante, a pesar de que el sistema mejora con cargas más grandes, la potencia de ruido se incrementa también.

Las diferentes hipótesis que se han planteado se centran en la fuente de alimentación que abastece al *opamp* implementado en la fuente de corriente Howland y en el propio *opamp* en sí.

Midiendo los ±45V generados en los convertidores DC-DC, se ha comprobado que existe un importante ruido de alta frecuencia en la señal de salida. Este resultado, invita a pensar que el valor de los condensadores usados en las salidas de los convertidores no es lo suficientemente grande.

Como posible solución, se ha propuesto añadir un filtro LC en la salida de los convertidores DC-DC. Mediante simulaciones en LTSpice, se ha podido comprobar la mejora que experimenta el sistema al aplicarle el filtro LC, en donde el inductor vale 1mH y el condensador 1μF.

Para el experimento, se ha modelado la fuente de ruido como una fuente de tensión sinusoidal de 800kH, 1Vpp y ±45V de tensión de offset.

En la Figura 3-2 se muestras los resultados de la simulación. A la izquierda se ha representado la corriente de salida del Howland ante una exitación sinusoidal de 200mV de amplitud y 30kHz. Para este caso, la carga empleada ha sido de 10kΩ. En la gráfica de la derecha, se representa la corriente de salida de la fuente de corriente Howland ante un pulso de valor 200mV. La carga empleada ha sido de 100kΩ.

A la vista de los resultados, es fácilmente comprobable que el filtro LC ayuda consideramente a filtrar las componentes de alta frecuencia de la señal. Puede notarse, además, lo que se indicó previamente: el ruido disminuye al aumentar el valor de la resistencia de carga.

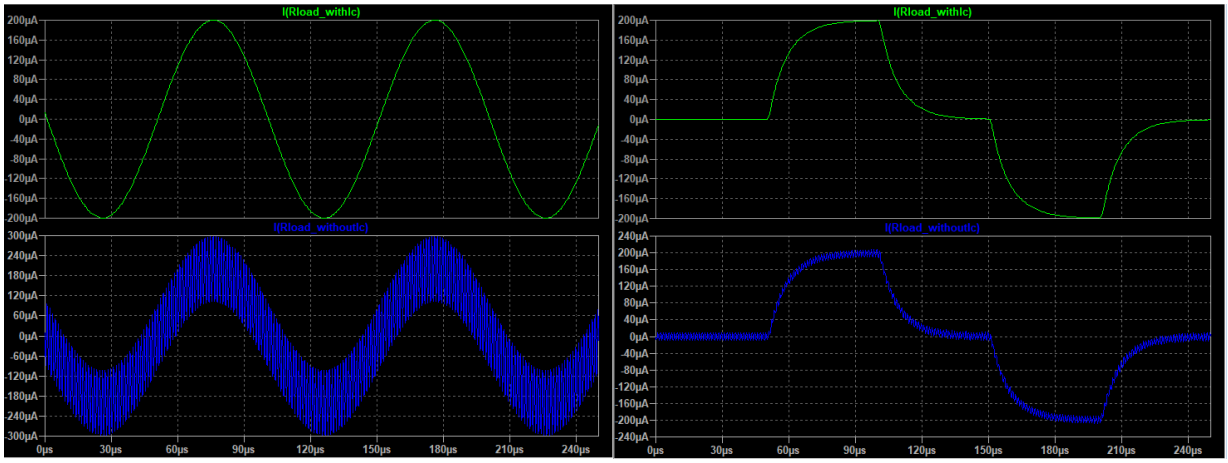


Figura 3-2. Resultados de la simulación transitoria de la salida del sistema cuando se le aplica un filtro LC en la salida de los convertidores DC-DC.

Con respecto a lo anteriormente mencionado, el incremento del ruido al emplear cargas pequeñas, se plantea también la posibilidad de que el amplificador implementado, al ser diseñado para trabajar en un rango de tensiones de hasta 100V, no se comporte adecuadamente para tensiones del orden de mV. Por ejemplo, para aquellos casos en los que se emplee impedancias del orden de 1kΩ y un rango de corriente comprendido entre 50-100µA, la tensión requerida abarcará entre los 50mV y 100mV. No obstante, se ha comprobado que, aunque mejoraba, para cargas mayores, también incrementaba su potencia. Por lo tanto, no es únicamente un problema que aparezca a tensiones bajas, sino que también está presente para valores mayores de tensión.

Como otra posible solución al problema de señales que presentan componentes ruidosas se plantea la posibilidad de elaborar un filtro pasivo a la salida del sistema. Este filtro, compuesto por impedancias complejas, se encargará de absorver las componentes de alta frecuencia sin modificar el valor de la corriente de salida que atraviese la carga. El filtro proporcionará una impedancia del orden de 10Ω para un rango de frecuencias hasta 50kHz y para frecuencias de 100kHz una impendancia del orden de 10MOhms.

En cuanto a las posibles mejoras que se pueden implementar en el diseño y funcionamiento del sistema neuroestimulador, se podría contemplar una distribución diferente en el *layout* de la PCB, con objeto de establecer una clara diferencia entre la zona aislada y la no aislada del circuito. De esta forma, se obtendría un único plano de tierra con una disposición de los elementos más concentrada, aportando más robustez a las señales contenidas en él frente a las interferencias. Otra posible alternativa se basaría en el empleo de *sockets* para los *opamps*, con el fin de poder ser reemplazados fácilmente.

En lo que respecta a la parte sotfware, podría desarrollarse una nueva implementación que incluya un procesamiento multi-hilo, tal que, una tarea se asigne a un *core* determinado y se encargue del procesamiento de los tiempos y otra tarea, asignada a otro *core,* se encargue de la ejecución de las muestras. Sería necesario, por tanto, que ambas tareas se encontraran perfectamente sincronizadas.

REFERENCIAS

[1] Wong, Y. T., Dommel, N., Preston, P., Hallum, L. E., Lehmann, T., Lovell, N. H. & Suaning, G. J. Retinal neurostimulator for a multifocal vision prosthesis. *IEEE TRANSACTIONS ON NEURAL SYSTEMS AND REHABILITATION ENGINEERING*, **15**, 425-434, (2007).

[2] Shepherd, R. K., Shivdasani, M. N., Nayagam, D. A., Williams, C. E. & Blamey, P. J. Visual prostheses for the blind. *Trends Biotechnol*, **31**, 562-571, (2013).

[3] Grossman, N., Bono, D., Dedic, N., Kodandaramaiah, S. B., Rudenko, A., Suk, H., Cassara, A. M., Neufeld, E., Kuster, N., Tsai, L. H., Pascual-Leone A. & Boyden, E. S. Noninvasive Deep Brain Stimulation via Temporally Interfering Electric Fields. *Cell*, **169**, 1029-1041, (2017).

[4] Wu, J., Dubhashi, S. & Bernstein, G. H. Inductive generation of arbitrary waveforms for electrical stimulation using implantable microcoils. *JOURNAL OF MICROMECHANICS AND MICROENGINEERING*, **14**, 1012-1021, (2004).

[5] Roya, L. A., Gunasinghab, R. M. K. D. & Rauck, R. New modalities of neurostimulation: high frequency and dorsal root ganglion. *Current Opinion in Anesthesiology*, **29**, 5, 590–595, (2016).

[6] Barriga-Rivera, A., d Elber, C., Paul, B. M. & Morley, J. W. An experimental setup for in vivo electrophysiological investigation in retinal prosthesis. *DRT4ALL*, 433-456, (2015).

[7] Todoroki, A., Shiomi, H., Mizutani, Y. & Suzuki, Y. Electrical Shorting between the Carbon-Fiber Cloth Electrodes of Structural Capacitors with a Glass-Fiber Cloth Separator. *Open Journal of Composite Materials*, **04**. 140-147, (2014).

[8] Kruchten, P. Architectural blueprints—The "4 + 1" view model of software architecture. *IEEE Software*, **12**, 42-50, (1995).

[9] Jiang, D. & Demosthenous. A. A Multichannel High-Frequency Power-Isolated Neural Stimulator With Crosstalk Reduction. *IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS*, **12**, 4, 940-953, (2018).

[10] Aquilera, O. A., Bayona, O. J. & Miranda, D. A. Criterios de Diseño de la Fuente de Corriente Howland. *VIS Ingenierias*, **6**, 1, 59-68, (2007).

[11] Mahnam, A., Yazdanian, H. & Samani, M. M. Comprehensive study of Howland circuit with non-ideal components to design high performance current pumps. *Measurement*, **82**, 94-104, (2016).

[12] Tucker, A. S., Fox, R. M. & Sadleir, R. J. Biocompatible, High Precision, Wideband, Improved Howland Current Source With Lead-Lag Compensation. *IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS*, 1932-4545, (2012).

[13] Pouliquen, P., Vogelstein, J. & Etienne-Cummings, R. Practical considerations for the use of a Howland

current source for neuro-stimulation. *IEEE Biomedical Circuits and Systems Conference*, (2008).

[14] Tibshirani, R., Kim, J., Lunde, R. & Todorova, S. Smoothing Splines. *Advanced Methods for Data Analysis*, (2014).

[15] Benzi, R., Sutera, A. & Vulpiani, A. The mechanism of stochastic resonance. *Journal of Physics A: Mathematical and General,* **14**, 11, 453-457, (1981).